第一章　绪　论

1.1　家庭电视游戏机

1.1.1　电视游戏

“电视游戏”，顾名思义，就是在电视机上玩的游戏。电视游戏的英文名称为“VIDEO GAME”或“TELEVISION GAME”。它是微电子技术发展的高技术产物。

玩电视游戏要通过电视游戏机才能实现。电视游戏机一般分为两大类：一类是大型电子游戏机，它采用 16 位或 32 位微处理器，游戏画面细腻逼真，音响效果较好，但它体积庞大、价格昂贵，多设在公共场所。另一类是家庭电视游戏机，它是面向家庭，以娱乐为目的的专用微型电脑。

1.1.2　家庭电视游戏机的起源

最早的家庭电视游戏机是美国设计的“雅达利”系列产品。它充分利用了 APPLE 微机（苹果机）的电子游戏软件，采用大规模集成电路、大幅度地简化整机结构，从而以较低的价格投放市场，迅速地进入家庭的娱乐领域。该机使用 65 系列八位微处理器，具有 4K 内存，游戏程序固化在可更换的游戏卡中，因而更换游戏节目简单、方便，一时成为家庭的娱乐珍品。较流行的节目有“运河大战”、“迷宫追车”、“滑雪”、“空战”、“猪八戒”等。国内常见的机型有“雅达利 2600”、“溢龙 7000”、“皇冠”、“汉龙”等型号。但是，由于该种游戏机内存容量小、游戏节目较简单、游戏动作较呆板、背景画

面较单调、而且操纵杆是手柄式，因而很快被后起之秀“任天堂”类游戏机所取代。

1.1.3 “任天堂”系列家庭电视游戏机

现在通常说的家庭电视游戏机，一般泛指“任天堂”系列的家庭电视游戏机。它是日本任天堂公司于1983年推出的家庭游戏机的换代产品。虽然它仍然使用65系列的八位微处理器和4K内存，但由于它采用了一个专门处理图象的大规模集成电路ppu，从而极大地提高了处理动画图象的能力。游戏节目的最小容量是24K，最大可以做到3M。它可以显示52种颜色，动画点阵最大为64×64点，背景分辨率达到256×240点，且有画面卷动功能。它的游戏动画功能已远远超过普通的微型计算机。同时，它还采纳了“雅达利”通过调换游戏节目卡更换节目的方法，整机和游戏卡分开出售，从而“任天堂”以生动活泼的动画图象、丰富精彩的游戏情节和低廉的销售价格，很快挤垮了“雅达利”而风靡全世界。

1987年，“任天堂”游戏机开始在我国的宁波、温州、广州等沿海城市销售。由于销路好、利润高，国内许多厂家纷纷与外商合资引进生产线进行生产。随着集成电路的大幅度降价，游戏机的价格急剧下跌，使得普通家庭都可以接受。九〇年以来，家庭电视游戏机在我国得到广泛的普及。

我国常见的机型有：台湾产的“胜天”、“小天才”、“小灵通”、“得其乐”；香港产的“海天使”、“爱迪”；南朝鲜的“爱丽”和国产的“天马”、“太山”、“终结者”等。这些机型都是“任天堂”游戏机的改型，所以仍属“任天堂”系列。

由于任天堂的游戏节目可以方便地从大型游戏节目中移植过来，因而它的节目源丰富、游戏精彩、涉及面广，迎合了男、女、老、中、青、少、幼不同的年龄阶层的爱好。常见的节目有“魂斗罗”、“绿色兵团”、“赤色要塞”、“沙罗曼蛇”、“超级玛丽”、“撞球”、“棋王”等，目前它已有数百个游戏节目，且以每年几十个新节目的速度递

增。丰富的游戏软件和完善的配套附件(软盘驱动器、键盘、光电枪等)赋予了“任天堂”强大的生命力,保持畅销不衰的好势头。

1.2 电视游戏机的理论基础

电视游戏机是现代科学技术的产物,是微型电子计算机技术和电视技术高度结合、发展普及的结果。无论是娱乐场所的大型游戏机,还是家庭拥有的电视游戏机和微型袖珍游戏机,其理论基础都根源于微型电子计算机。为此,本节先熟悉了解一下电子计算机的基础知识。

1.2.1 电子计算机的组成

为了便于了解计算机的组成,我们先从人们是怎样用算盘算题谈起。

小明期末考试语文成绩 93 分,数学成绩 99 分,常识成绩 86 分,地理成绩 78 分,求平均成绩是多少。

计算时,一般先将算式写在纸上:

$(93+99+86+78)\div 4=?$

演算开始,首先进行连加运算

$93+99+86+78$

得出总成绩为 356。接着按除法口诀计算

$356\div 4$

的商等于 89。即平均成绩为 89 分。把答案写在纸上,至此,计算过程结束。

在上面的计算过程中,参加算题的有纸、笔、算盘和人。人们从自己的计算实践中得到启示,模仿自己的计算过程而创造了电子计算机。

电子计算机也具有与上述计算过程类似作用的几个组成部分:

• 能进行数字运算的“算盘”——运算器。

• 能记录和保存原始数据、运算步骤及中间运算结果、最后计算结果的“纸”——存储器。

• 能书写数据、算式和计算结果的“笔”——输入、输出设备。

• 能控制“算盘”、“笔”和“纸”协调工作的“指挥者”——控制器。

这样，运算器、存储器、输入设备、输出设备和控制器就成为电子计算机的五个主要组成部分，它们之间的联系如图 1－1(a)。

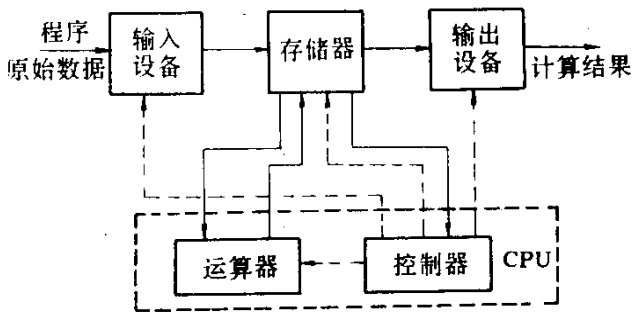


图 1－1(a)　电子计算机组成框图

实际上，电子计算机除了图 1－1(a)所示的五部分外，还必须有电源等设备。

1.2.2　运算器

运算器是直接完成各种算术运算和逻辑运算的装置。算术运算就是加、减、乘、除等运算；逻辑运算就是按照逻辑代数规律进行的运算，如逻辑加、逻辑乘等。此外，运算器还能做一些别的动作，如数码的传送、移位以及给出转移特征等。所有这些运算和动作统称为操作。指挥机器进行操作的命令，称为指令。一台机器所具有的指令的全体，称为指令系统。

机器的各种运算可以由相加及移位这两个基本操作来实现。为了进行运算，参加运算的数需要在运算器中暂时存放起来，称为

寄存。每次运算所得的中间结果也要暂时保留，使上次运算结果能直接为下一次操作所利用。因此，运算器必须具备三个最基本的功能：

1. 数码的寄存。

2. 数码的移位。

3. 数码的相加。

寄存数码的部件称为寄存器。在寄存器里增加一些附加电路，便可实现数码的移位。而数码的相加，则要用到加法器。所谓加法器就是直接实现数码相加的部件。

由此可见，运算器就是由加法器以及若干个寄存器组成的。

1.2.3 存储器

存储器是存放数据和程序的装置。不论是数据或指令都是一些用二进制数表示的代码。存储器的基本功能是能够把许多的代码，按需要存进去(写入)或取出来(读出)。这种功能可比喻为人的“记忆”，所以，存储器也叫记忆装置。

由于存储器要存储大量的代码，因此它的内部分成许多小单元，正如一座大宾馆有许多房间一样。存储器内的小单元，称为存储单元。每个存储单元有一个编号，正如宾馆中每个房间均编有房号一样。存储单元的编号，称为地址。每个存储单元通常存放一个有独立意义的代码，即存放作为一个整体来处理或运算的一组数字。这样的一组数字称为字。字的长度，即字所包的位数，称为字长。我们所说的任天堂使用 65 系列的八位微处理器，就是指它的字长是八位。存储器所具有的存储单元的总数，称为存储容量(简称容量)。存储容量越大，意味着机器的功能越强。例如，我们说某计算机的存储容量为 8192 个单元，意思就是说能记忆 8192 个数，一般简记为 8K。这里 1K 字等于 1024 个字。我们说“任天堂”有 4K 内存就是指它有 4096 个存储单元。

存储器按其在计算机中的作用分为两类：一类称为内存储器，

简称内存；另一类称为外存储器，简称外存。内存容量比外存少，但存取速度快。对游戏机而言，4K 为内存，游戏卡则为外存。

1.2.4 输入、输出设备

计算机和外界联系要通过输入、输出设备才能实现。输入、输出设备也叫外部设备，简称外设。输入设备的主要功能是将原始数据、程序等变换成计算机所能识别的二进制形式的电信号，并把它送入存储器中；而输出设备则是把计算结果或中间结果以人们能识别的各种形式，如数字、字母、符号或图形、表格等表示出来。因此，输入、输出设备实现了人和计算机之间的信息交流，是计算机的重要组成部分。

输入、输出设备的种类和形式很多。常见的输入设备有纸带输入机、卡片输入机、键盘、光笔以及模数转换器等。目前正在研制手写体符号输入和语言声音输入装置。输出设备有纸带穿孔机，卡片穿孔机、行式打印机、X－Y 记录仪、绘图机和显示器、数模转换器等。理想的输出设备应该“会写”、“会讲”。“会写”现已圆满地做到，“会讲”即声音输出，目前尚在研究中。

1.2.5 控制器

控制器是整个机器的指挥系统。它通过向机器的各个部分发出控制信号来指挥整台机器自动地、协调地进行工作。但控制器到底是怎样指挥机器工作的呢？实际上，控制器是根据人们事先编好的程序来进行控制的。计算机先做什么、后做什么、如何处理可能遇到的一切情况，都要由程序来决定。人们把事先考虑好的意图表达在程序中，而控制器则按照程序的规定去指挥机器工作。因此，可以说控制器是按照人的意图（由程序体现）来指挥机器工作的。计算机自动工作的过程，实质上就是自动执行程序的过程。

运算器、控制器和存储器合称为电子计算机的主机。其中运算器与控制器合称为中央处理机，简称 CPU（CPU 是英文 Central Processing Unit 的缩写）。而输入设备、输出设备以及外存储器合称

为外部设备。一台主机所能配备的外部设备的品种、数量，是根据机器的功能大小和使用要求而定的。现在我们再画出一种电子计算机组成框图如图 1—1(b)，可能有且助于读者加深印象。

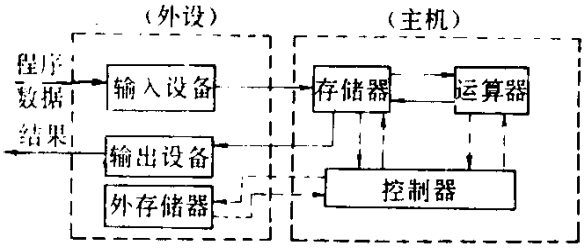


图 1—1(b)　电子计算机组成框图

1.2.6　电子计算机的特点

同其它计算工具相比，电子计算机有如下显著特点：

1.　计算速度快

一个人如果用算盘进行计算，每天工作八小时，一般能完成几千次运算，平均每秒钟运算不到一次。可是一般的电子计算机的运算速度每秒已达几百万次、几千万次、甚至几亿次。一台每秒运算一百万次的计算机在一分钟内完成的计算量，就相当于一个人用算盘工作几十年的计算量。如计算圆周率 π 的精确值，著名数学家契依列曾耗 15 年之精力才计算到小数点后 707 位。而用普通计算机仅几分钟就可计算到 1000 位以上。

2.　计算精确

计算机的计算精度取决于字长。人们可以根据实际需要来设计计算机的字长。字长越长、运算结果越精确。一般选八位到几十位。

3.　可靠性高

人们使用算盘进行计算，大脑有疲劳的时候，疲劳后则容易出错。而对于计算机来说，在数据输入、程序输入时，可反复检查，再启动运算；机器本身由于采用了大规模或超大规模集成电路，使计算机的连续无故障运行时间达几万、几十万小时以上，也就是说可

以连续几个月、甚至几年不停地工作而不出错误，因而可靠性高。

4. 自动化程序高

计算机从正式开始工作，到送出工作结果，整个工作过程都是在程序控制下自动进行的，完全用不着人去参与。这样，采用计算机可使人们从繁琐的、重复的脑力劳动和体力劳动中解放出来，把精力用在创造性的劳动上。

计算机之所以自动化程度高，基本因素之一就是它具有“记忆”能力。计算机不仅可以存储原始数据、中间结果和最后结果，而且更重要的是可以存储程序。存储程序是电子计算机的一个重要工作方式，是机器能自动进行计算的基础。游戏机正是利用了这一特性，把游戏节目的程序存储在外存（游戏卡）中，使用时只要联接存储器（插上游戏卡），接通电源就可玩电视游戏。

另一个基本因素是具有逻辑判断力和选择能力。它可以进行各种逻辑判断。如对两个信息进行比较，根据比较的结果，自动确定下一步该做什么。这种能力使计算机能够完成各种计算任务，进行各种过程控制和对有关数据进行处理。以“下棋”游戏来说，游戏者在不同的位置下一子，电脑也会在相应的位置下一子，你变，它也变，计算机充当了一名相当高明的对手。

5. 通用性强

计算机可用于数值计算、数据处理、自动控制、辅助设计、逻辑关系运算与人工智能等方面。计算机的应用范围已经渗透到各行各业，尤其是微型电子计算机以体积小，性能高、成本低、售价廉的优势已逐步占领了技术、经济、企业、家庭等各个领域。极强的通用性为电子计算机应用的推广、普及开创了广阔的前景。

1.2.7 电子计算机中数的表示方法

人们习惯于十进制的计数方法。用0～9这十个数字的不同组合，可以表示任意一个数。在十进制中共有十个基数，由低位向高位的进位是逢十进一。电子计算机中采用的是二进制，它只有两个

基数，即 0 和 1，由低位向高位的进位是逢二进一。非十进制方式，初看起来似乎很别扭，其实也不陌生。在我们的日常生活、工作中也并不全使用十进制。例如：1 小时等于 60 分钟、1 分等于 60 秒，是 60 进制，逢 60 进一。

为什么电子计算机偏偏采用二进制而不是其他进制呢？这主要是因为二进制只有两个基数，便于用物理元件的两种不同的稳定状态来表示。例如：电位的高、低；开关的通、断；电灯的亮、灭；脉冲的有、无等，都可以用来表示二进制数 1 和 0。显然，这种物理元件在技术上是较易实现的。若采用十进制，就要求一种有十个不同稳定状态的物理元件，这是十分困难的。既使制造出来也必定很复杂，那么造出来的计算机也必然很庞大。另外，二进制数的运算规则比较简单，也是电子计算机采用二进制的一个原因。

二进制虽然有以上优点，但它的位数较多，书写和识读都很不方便。例如十进制数 15 用二进制写出来就是 1111；十进制数 117 仅有 3 位，而用二进制表示为 1110101，则有 7 位。再一点就是人们也不习惯于使用二进制进行数字计算。为解决人们习惯十进制、计算机适合二进制的矛盾，人们采取了翻译解决的方法。即：在操作计算机时，人们仍按十进制编制程序和给出数据，输入计算机后，由计算机自动将其转换为二进制数送入内存；计算完毕输出时，再把二进制数结果翻译成十进制数显示在屏幕上或打印在纸上。这两次翻译过程全部由计算机自动完成，所以对使用者来说是十分方便的。

另外，计算机中还经常使用十六进制。十六进制共有十六个基数，从低到高依次是 0、1、2、3、……9、A、B、C、D、E、F。表 1－1 给出二、十、十六进制数对照情况。

表 1—1　二进制、十进制、十六进制数对照表

二进制	十进制	十六进制	二进制	十进制	十六进制
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

1.2.8　二进制数的运算规则

电子计算机中数的运算以二进制的算术运算为基础，运算规则很简单。下面简略介绍它的四则运算。

1.　加法

二进制的加法规则是：

0+0=0

0+1=1

1+0=1

1+1=10

例 1.1　求 38+10=？

	十进制	二进制
被加数	38	100110
加　数	+1	+　1010
和	48	110000

2.　减法

二进制的减法规则是：

0−0=0

1−0=1

1−1=0

10−1=1

例 1.2　求 48−38=？

	十进制	二进制
被减数	48	110000
减　数	−38	−100110
差	10	001010

3. 乘法

二进制的乘法规则是：

$0\times0=0$

$0\times1=0$

$1\times0=0$

$1\times1=1$

例 1.3 求 $14\times5=?$

十进制

被乘数		14
乘　数	×	5
积		70

二进制

			1	1	1	0
×					1	0	1
			1	1	1	0
		0	0	0	0	
	1	1	1	0		
积	1	0	0	0	1	1	0

从例 1.3 可见，做二进制乘法时，若乘数为 1，则被乘数照写一遍，其最右边的一位应与相应的乘数对齐；若乘数为零，实际上没有什么作用(但向左移了一位)。所有乘数乘完后，把各部分乘积加起来即求得结果。可见，二进制乘法实质上是由加法和移位两种操作来实现的。

4. 除法

二进制除法运算是乘法的逆运算，它是由减法和移位两种操作来实现的。

总之，二进制数的四则运算可归结为加法、减法和移位三种操

作。实际上，在计算机中往往不直接进行减法运算，而是用加负来代替减正。这样一来，四则运算就简化为加法和移位两种操作了。

在电子计算机中，怎样表示数的正负，怎样用加负数的方法来代替减法呢？现在进行简单介绍。

例 1.4 求 70÷5=？

十进制

```
   14
5 ) 70
    5
   ---
    20
    20
   ---
     0
```

二进制

```
          1110
101 ) 1000110
      101
      -----
       11110
       101
      ------
        1010
        101
       -----
           0
```

电子计算机中表示数的正负不是在数的前面加“+、−”号，而是在二进制数前面增加一位“符号位”。当“符号位”为 0 时，表示此数为正；当“符号位”为 1 时，则表示此数为负。例如在八位微机中 +48 和 −38 写为二进制形式为：

$(+48)_{10}=0.0110000$

$(-38)_{10}=1.0100110$

其中圆点前面的 0 和 1 就是符号位，圆点后面的为数据。（注：圆点是为便于阅读和区别而人为添加的，实际上没有圆点。）

加负数是通过加补码的方法来实现的。什么叫补码呢？对于一个二进制负数来说，可以有三种不同形式的编码，即原码、反码补码。

- 原码——原码保持了数的原来形式，即数的数据部分不变只是符号位根据数的正、负取“0”或“1”。

- 反码——反码就是将数的数据部分各位取反，即“0”换代“1”，“1”换代“0”。如－38的反码为1.1011001。
- 补码——补码就是反码的最低位加1。如－38的补码为1.1011010

若引入变量X，设X的原码记为[X]原、X的反码记为[X]反、X的补码记为[X]补，则有

$X=-38$

$[X]_{原}=1.0100110$

$[X]_{反}=1.1011001$

$[X]_{补}=1.1011010$

下面以＋48－38＝?为例说明怎样用加补码的办法实现减法。

＋48的二进制数原码	0.0110000
－38的二进制数补码	＋ 1.1011010
和	10.0001010

对于八位微机，符号位为最高位，再往前的进位由于没有设备保证而丢失。因此，上例的计算结果为

0.0001010

即十进制的10，显然结果是正确的。

当两数相减的结果为负数时，上述方法仍然适用，只不过运算的结果为补码，应将其变为原码才得正确的结果。

例1.5 求38－48＝?

＋38的二进制原码	0.0100110
－48的二进制补码	＋ 1.1010000
结果为二进制补码	1.1110110

已知补码求原码，必须将补码减1再取反。减1的操作仍可使用加1的补码。则有

结果的二进制补码		1.1110110
−1的二进制补码	+	1.1111111
结果的二进制反码		11.1110101
把反码变为原码		1.0001010

所得结果为1.0001010，即十进制数−10，显然也是正确的。

通过上述示例，读者可能觉得这样做运算次数多、太麻烦、运算效率低。诚然，若手工计算，用这种方法确实太笨。但是这“笨方法”的运算操作简单。而电子计算机最擅长的就是进行这方面的工作。它每秒钟可运行几百万到几亿次，因此计算机就可以“笨中取巧”，用最简单的操作完成复杂的运算。

1.2.9 基本电路常识

在电子数字计算机中通过什么样的电路来实现二进制数的运算呢？如前所述，二进制数只有两个基本数字0和1，因此，对二进制数的运算实际上可归结为对0和1的运算。0、1可以和逻辑关系中的是和非相对应，所以对0和1的运算也就是对是和非的运算，这种运算实际上就是一种逻辑判断。电子计算机中，就是用一些能进行逻辑判断的电路来实现对二进制数运算的。

能够对0和1进行基本逻辑运算的电路，称为基本逻辑电路。常用的基本逻辑电路有：或门逻辑电路、与门逻辑电路、非门逻辑电路和双稳态触发电路等。下面就对这些电路作一简单介绍。

1. 或门逻辑电路

我们先从人们最熟悉的电灯线路开始，见图1−2。

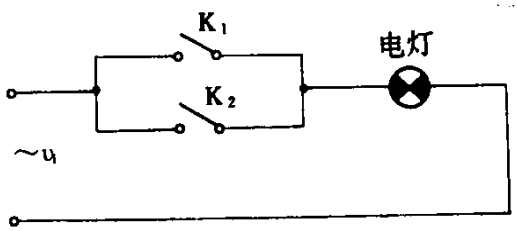


图1−2 “或”现象电路

在图 1－2 中，开关 K1 和 K2 并联后再和电源及电灯串联。我们约定灯亮为 1 状态，灯灭为 0 状态。开关状态和电灯状态的对应关系如表 1－2。

表 1－2　开关通断与电灯亮灭的关系

开关状态		电灯状态	说　明
K1	K2		
0	0	0	K1、K2 都断开，灯灭
0	1	1	K1 断、K2 合，灯亮
1	0	1	K1 合、K2 断，灯亮
1	1	1	K1、K2 都合，灯亮

如果把开关状态称为输入，电灯状态称为输出，则表中所列的四种现象所代表的输入和输出之间的关系就可以表示为：

0＋0＝0

0＋1＝1

1＋0＝1

1＋1＝1

以上等式就是逻辑加法（或）的关系式。这与二进制加法规则是很相近的，除 1＋1＝10 与逻辑加法中的 1＋1＝1 不同外，其余三式完全相同。

在电子计算机中，能够实现逻辑加法的电路称为或门逻辑电路，简称或门。图 1－3 所示是一种最简单的或门电路。

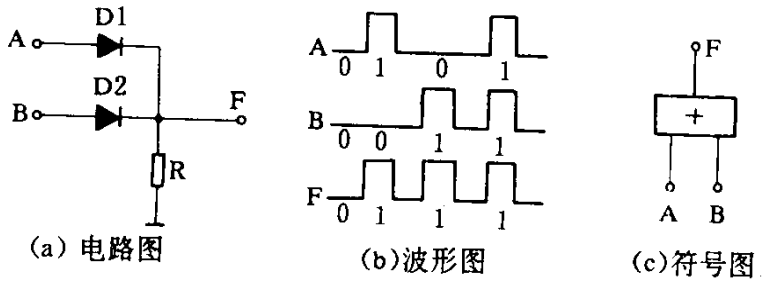


图 1－3　或门逻辑电路

图 1－3(a)中 D1、D2 为二极管，A、B 为输入端，F 为输出端，R

为电阻。约定：输入端A、B为高电位时代表1状态，低电位则代表0状态。同样，输出端F为高电位时代表输出为1状态，低电位则代表输出为0状态。当A和B中有一个为高电位时，电流通过二极管经电阻流向地，在电阻R上产生电压降，使输出F端为高电位；当然，A和B都是高电位时，F也是高压位；当A、B都是低电位时，电流被D1、D2阻挡，没有电流通过，R上没有压降，这时F为低电位。输入、输出端的电位关系用波形图表示如图1－3(b)所示。另外输入端的数目可以超过2个而有3个、4个或更多。输出与输入的关系，可写为一般表达式：

$$F=A+B+C+D+E+\cdots\cdots$$

在电子计算机逻辑图上，或门逻辑电路常用图1－3(c)所示的符号表示。矩形内的“＋”号表示逻辑或。

2. 与门逻辑电路

图1－4表示了“与”现象的电灯线路。

表1－3表示了开关通、断与电灯亮、灭的对应关系。

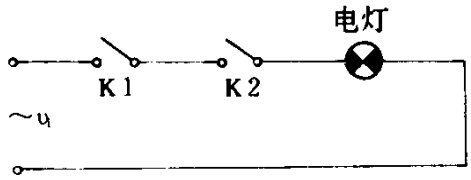


图1—4 “与”现象电路

与门逻辑电路输入和输出的关系表示如下：

$0\times0=0$

$0\times1=0$

$1\times0=0$

$1\times1=1$

这就是逻辑乘法，与二进制乘法规则完全相同。

表 1—3　开关状态与电灯状态的关系

K1 状态	K2 状态	电灯状态	说　明
0	0	0	K1 断、K2 断，灯灭
0	1	0	K1 断、K2 通，灯灭
1	0	0	K1 通、K2 断，灯灭
1	1	1	K1 通、K2 通，灯亮

在电子计算机中，能够实现逻辑乘法的电路称为“与门”逻辑电路，简称与门。最简单的与门电路如图 1—5 所示。电路原理留给读者分析。

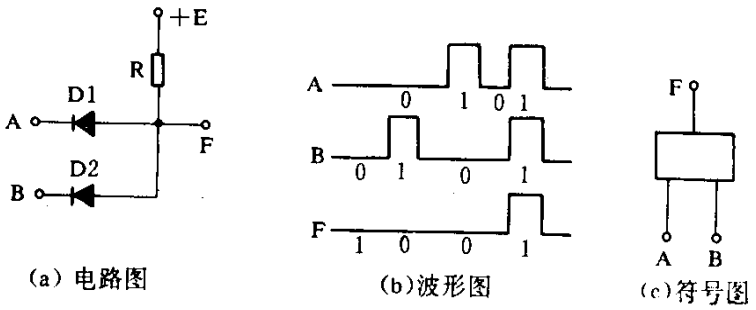


图 1—5　与门逻辑电路

3.　非门逻辑电路

非门逻辑电路可用图 1—6 所示电灯线路予以说明。并关状态与电灯状态的对应关系如表 1—4。由图、表可见，当 K 断开时，电灯亮；当 K 合上时，电流从 K 中流过，电灯两端无电位差，故电灯灭，所以开关 K 对电灯起到了否定的作用，一般称为“非”。若输入状态记作“A”，则输出状态记作“$\overline{A}$”，读作“非 A”。

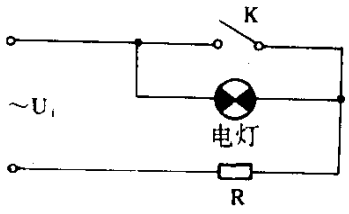


图 1—6　“非”现象电路

表 1—4　开关状态与电灯状态的关系

开关状态	电灯状态	说　明
0	1	K 断，灯亮
1	0	K 合上，灯灭。

电子计算机中，能够实现“非”这一逻辑关系的电路称为非逻辑电路，简称非门，通常又叫做反相器。图 1—7 为反相器的一种典型电路。当输入端为高电位时，三极管 BG 导通，电流由 +E 流经电阻 R_C 和 BG 到地，由于 BG 导通时内阻极小，输出端 $\bar{A}$ 为低电位，即 A 为“1”，$\bar{A}$ 为“0”。反之，A 为“0”时，BG 截止，+E 通过 R_C 使 $\bar{A}$ 为“1”。在这里，BG 起着开关的作用。非门的符号如图 1—7 (c)所示，矩形框上面的小圆圈表示“非”。

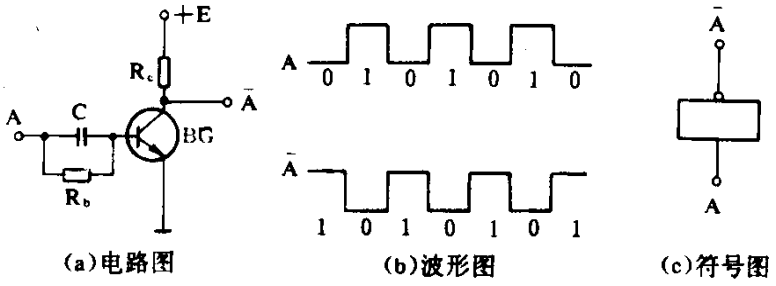


图 1—7　非门逻辑电路

4.　触发器：

触发器是一种存储器件，它以低电平或高电平的方式储存一位二进制数。

一个与门和一个非门接在一起就组成了一个复合门电路，称为“与非”门。与非门的逻辑功能为：只有当输入端均为高电位时，输出才是低电位；否则，输出高电位。与非门的符号如图 1—8 所示。

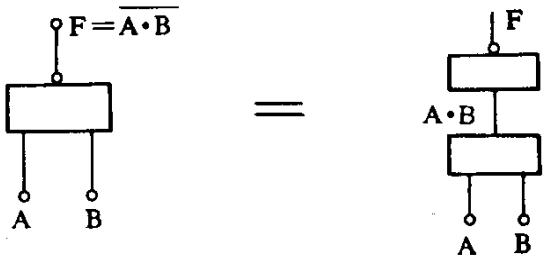


图 1—8 “与非”门符号

用两个与非门可以组成一个最简单的触发器，称为 R—S 触发器，如图 1—9 所示。图中 R 和 S 为输入端。其中，R 为置“0”输入端，S 为置“1”输入端。在没有脉冲输入时，R 和 S 端均为高电位。R—S 触发器有两个输出端，即 Q 端和 $\bar{Q}$ 端。当 Q 为高电位时则 $\bar{Q}$ 端为低电位；当 Q 端为低电位则 $\bar{Q}$ 端为高电位。

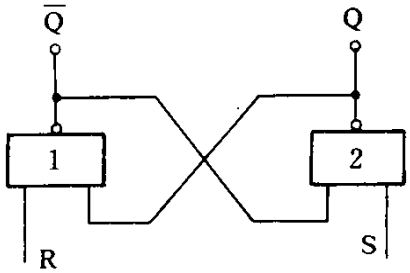


图 1—9 基本 R—S 触发器

R—S 触发器的工作原理如下：

a. 没有触发脉冲输入时 R=“1”、S=“1”，即 R、S 端均为高电位。假定这时 Q=“1”则 $\bar{Q}$=“0”。

b. 当 R 端有触发脉冲时，即 R=“0”、S=“1”，由与非门的特性可知 $\bar{Q}$=“1”，这时与非门 2 的两输入端均为 1，则有 Q=“0”。所以说 R 是置“0”端。现在，即使 R=“1”，也仍保持 Q=“0”的状态。

c. 当 s 端有触发脉冲时，R=“1”，S=“0”，与非门 2 有一个输入端为低电平，故必然有 Q=“1”。现在与非门 1 两输入端均为高电平，故有 $\bar{Q}$=“0”。即使 S 端的触发脉冲撤去，也仍然保持这种状态。所以把 S 称置“1”端。

可见，只要控制 R、S 端的触发脉冲，就可以改变触发器的输出端 Q 的电位。也即是控制触发器储存“0”或“1”。不难看出，一个 R—S 触发器可以储存一位二进制数。

从以上的介绍中，我们看到，与门、或门、非门是判定器件，使用它们可以完成算术运算或逻辑运算。触发器是存储器件，使用它可以组成存储器，把各种数据、信息储存起来。若再加上一些辅助电路和设备，就可以组成一台完整的电子计算机。

总而言之，现代电子计算机虽然复杂而神奇，但归根结底，它只能运算、储存 0 和 1 两个数，而它们基本电路也极为简单。问题在于有了这些积木块，就看你如何利用它们去建造小别墅或摩天大楼了。

电子计算机理论是一门专业课程，本节仅进行了最简单的介绍。对于不具备无线电常识的朋友，理解本节内容可能有困难。但是不要担心，读者可以跳过本节内容，甚至跳过本章内容。对于学习 F BASIC 语言来说，即使丝毫不懂计算机理论也影响不大，当然，若能掌握计算机理论知识将更有助于 F BASIC 语言的学习和提高。

1.3 电视游戏机的工作原理

上一节我们曾谈到电视游戏机是微型电子计算机技术和电视技术高度结合的产物，其理论基础根源于电子计算机，这说明游戏机与计算机有着密切的联系。本节将从游戏机的结构、工作过程入手，讨论游戏机变为计算机的可能性。

1.3.1 游戏机的结构

游戏机的结构框图示于图 1—10。

• 整流电源把 220V 的交流电压变压整流后变为 12V 的直流电压供游戏机使用。

• 电脑板是游戏机的心脏，它主要由中央处理器CPU、图象处理器PPU、输入、输出接口电路和内存储器组成。

• 游戏卡用来存放游戏程序

• 调制器用来把由电脑板输出的视频信号调制为射频信号，供电视机接收显示。

• 操纵器用来控制游戏的开始、选项，及游戏中主人翁的各种动作。

• 手枪是集光、电、机械于一体的控制器，同操纵器一样，也可控制游戏进行。游戏者可以拿手枪瞄准屏幕做射击游戏。

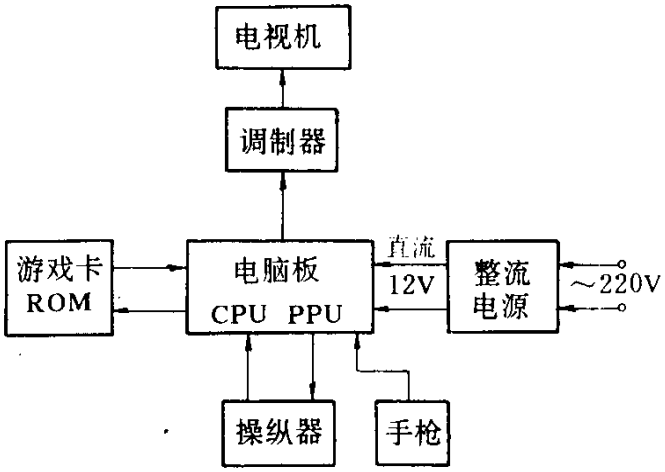


图 1－10 游戏机结构框图

1.3.2 电视游戏机的工作过程

接通电源后，主机复位，电脑板上的微处理器CPU开始执行节目卡ROM中的游戏程序，并同时处理由操纵器输入的串行数据。CPU把图象数据送给视频处理器PPU，由PPU把它转换为PAL制的电视信号送入调制器，调制器则把视频信号和由CPU送来的音频信号合成为全电视信号并调制成确定频道的射频信号送入电视机，或者把视频、音频信号直接送入监视器和音频放大器。

在游戏中，游戏者手、眼、脑并用，根据画面的变化，不断发出各种指令信号(操纵游戏)，从而完成游戏的全过程。

1.3.3 游戏卡的工作原理

游戏节目的变换方法是调换游戏卡。游戏卡主要由只读存储器(ROM)构成。存储器中存放着整个游戏的逻辑过程(即监控程序)和全部游戏图象的数据。当人们操纵游戏机的操纵器时,CPU便会根据操纵指令按游戏所规定的逻辑确定游戏的发展。存储器存储程序的方式犹如旅馆接待安排投宿的客人。当我们访问旅馆中的某位客人时,只要知道他的房间号码就可以直接找到他,而不必从第一个房间开始依次向后查找。同理,CPU 执行存储器中的监控程序就如访问客人一样,只要知道程序的存储地址,就可以直接访问它。因此,CPU 可以根据游戏的进程迅速地调用有关程序,以保证游戏能够连续不断地玩下去。

1.3.4 游戏机功能的扩展

我们若去掉图 1－10 中的手枪和操纵器,把电视机作为输出设备,游戏卡作为外存储器,那么与图 1－1(b)相比,可以看出图 1－10 仅比图 1－1(b)少了输入设备。因此只要配上输入设备,游戏机就变成一台完整的计算机了。

与游戏机配套的常用输入设备是键盘。如上所述,游戏机配上键盘就变成了计算机。我们把这种计算机叫做"家庭电视电脑游戏学习机",简称"电脑游戏机"(也称为键盘游戏机、电脑学习机等。笔者认为电脑游戏机较贴切一些)。实际上,任天堂游戏机的正式名称就是"FAMILY COMPUTER",即"家庭计算机"的意思。

1.4 电脑游戏机

目前,市面上有多种型号的电脑游戏机,有键盘、学习卡分离式,也有游戏机、键盘、学习卡装在一起的组合机。常见的键盘有:"天马 TM5510"、"天津 KB200"、"小天才 IQK－91"、"BIT－79"、"大岛"、"裕兴"等型号,常见的组合机有"科特 FCS－90"。这些机

型虽然型号不同、式样各异，但其功能基本上都是相同的。它们都有104个背景图形、16个卡通图案、52种色彩，都能够进行算术运算、谱曲演奏、打字练习、背景绘图等操作，都使用F BASIC语言进行程序设计。所不同的仅是由于移植的版本不同，各种机型能够识别的F BASIC指令条目数量多少稍有差别。比较几种机型，只有裕兴中西文二合一卡和FCS－90的F BASIC系统较完善(FCS－90的F BASIC版本有两个，一个为BS. 2A、一个为V3. 10，其它机型的多为V3. 1或V3. 3。裕兴还可以中文编程。一般的BS卡的F BASIC版本仅为BS. 2A，其编程功能要差一些。)为使介绍的F BASIC语言最系统、最全面、最有代表性，本书以FCS－90机为例进行介绍，其它机型的功能则全部囊括其中(不包括中文功能)。

1. 4. 1 电脑游戏机的构成

电脑游戏机的构成见图1－11。

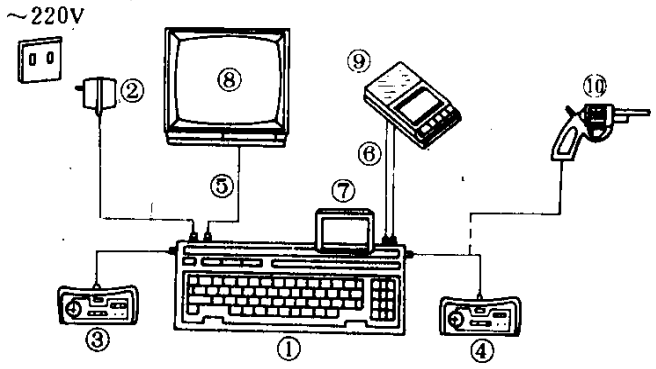


图1－11 电脑游戏机的构成示意图

①主机

②直流电源

③Ⅰ号操纵器

④Ⅱ号操纵器

⑤同轴电缆连接线(75Ω)

⑥数据信号传输线。

⑦游戏卡

⑧电视机

⑨盒式录音机

⑩光电枪

1.4.2 主机结构

主机结构见图 1—12

①Power 电源开关

②功能键

③字符/数字键

④主机机架

⑤方向键

⑥Ⅱ号扩充口(Ⅱ号操纵器插口)

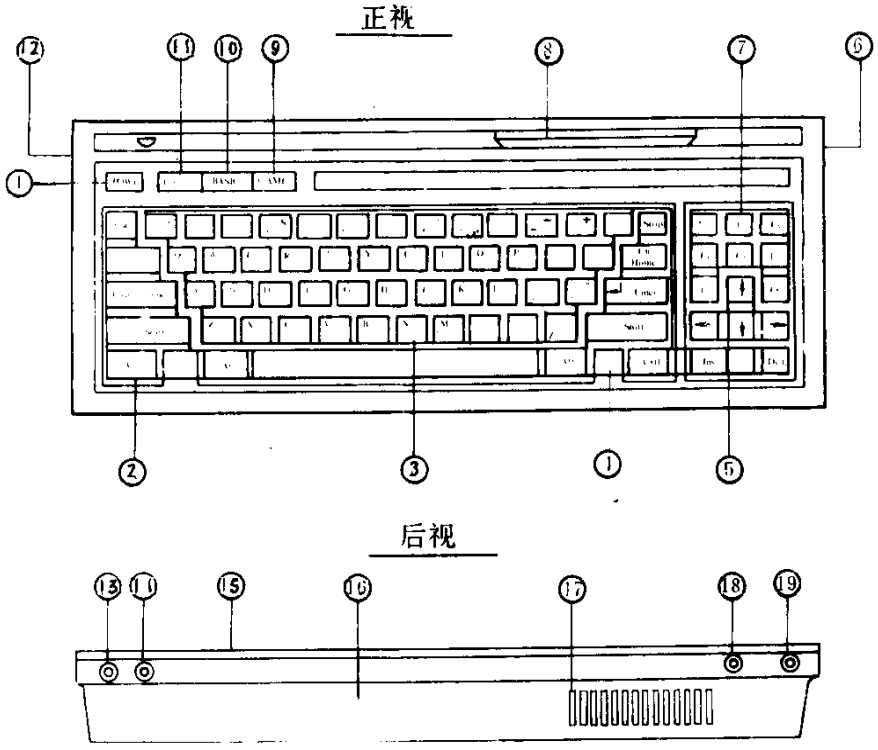


图 1—12 主机结构图

⑦特珠功能键

⑧游戏卡插口

⑨GSME(游戏)状态指示灯

⑩BASIC(F BASIC 程序设计)状态指示灯

⑪BS(学习)状态指示灯

⑫I 号扩充口(I 号操纵器插口)

⑬SAVE 数据输入插座

⑭LOAD 数据输入插座

⑮主机上盖

⑯主机底座

⑰散热孔

⑱RF(射频)输出插座

⑲DC(直流)电源插座

1.4.3 电脑游戏机的特点

FCS—90 计算机系统集学习与游戏功能于一体,设计科学、合理,特别适合青少年朋友学习电子计算机知识和家庭娱乐。主要特点如下:

1. 学习、游戏功能强

本机的学习、游戏功能包括:

- 算术运算。运算能力为 8 位有效数字,取值范围 0.0000001～99999999。
- 音乐谱曲与演奏。(可演奏三重奏乐曲)。
- 打字练习(英文打字)/留言。
- BASIC 程序设计。
- 利用背景图形绘画。
- 自编游戏程序。
- 电视游戏,可使用任天堂、小天才游戏卡。
- 绘制卡通图片。

- 数据资料的读/写等。

2. 设计科学、合理

采用模块组件结构紧凑、科学、合理。标准键盘可适应不同年龄人员操作。

- 内存 72K×8 系统软件。
- 配备增强型 F BASIC V3.10 语言系统。
- 内设 BS 学习系统，配备 F BASIC BS.2A 语言系统。

3. 采用标准键盘，硬件可靠

- 配 73 键电脑键盘。键位按国际通用标准排列，与普通英文打字机一致。
- 操作简单、方便、灵活。
- 采用高科技 CPU、PPU 超大规模集成电路(VLSI)并行工作，集中控制，工作稳定、可靠。

4. 采用电视机做终端显示

设有电视信号 RF(射频)输出口，可用 PAL/D 制式彩色电视机或黑白电视机做终端显示。

5. 可写/读资料

配有数据输入/输出接口，可外接盒式录音机，储存或读取所需资料。

6. 电视游戏

- 配有 60 针游戏卡插座，适用容量为 24K～10M 的各种“任天堂”游戏卡。
- 设有自动弹簧式防尘盖，可降低灰尘污染，延长使用寿命。

7. 外设件扩充性能强

附设两个 15 芯专用插座，便于连接操纵器、光电枪、立体镜等多种外设。

8. 工作状态指示灯可自动显示电脑系统所处的工作状态。

9. 使用组合按键可方便地选择功能和转换计算机的工作状

态，使转换工作简便、灵活。

10. 主要技术指标：

- 八位 CPU、PPU 微处理器。
- 内存容量：12K×8 RAM，其中 8K×8 供用户使用。
- 系统软件：72K×8 ROM，固化 F BASIC 及 BS 基本系统。
- 显示器接口：RF(射频)，输出电平＞70 dB。
- 盒式磁带接口：传输速度 1200 波特。
- 显示能力：

  字符：28 字符×24 行(BS、BASIC)

  32 字符×28 行(游戏)

  32 字符×30 行(底背景)

  字符结构：8×8 点阵

  卡通结构：8×8 点阵、16×16 点阵

  颜色：52 种

  解析度：256×240 点
- 声音效果：方波　　2 个

  三角波　1 个

  效果音　1 个
- 电源电压：交流 220V±15%，50/60 Hz
- 环境温度：0～40℃
- 相对湿度：40～80%
- 功　　耗：5W

1.5 电脑游戏机的使用

本节重点介绍屏幕编辑方法、算术运算、谱曲与演奏、键控绘图和数据资料的写/读操作等功能的使用。

1.5.1 开机操作

1. 与电视机的连接

使用附带的 75Ω 同轴电缆连接线连接电视机天线端子 VHF 和学习机的 RF 输出插座。

2. 连接电源

• 将直流稳压电源的圆形 DC(直流)电源插头插入学习机的 DC 电源插座。

• 将直流稳压电源的扁形交流电源插头插入市电(交流 220V)电源插座。

3. 开机

确认以上接线正确无误后,打开电视机,按下学习机电源开关,这时"BS"状态指示灯亮,系统即自动进入"BS"学习功能状态。

4. 调试

将电视机上的频道转换开关调试至 VHF 的 11 频道,调整电视机使能收到如图 1—13 的调试画面,并且图像清晰、无杂音。在调 试画面上,学习机会输出一个拥有 52 种彩色变换的图案,可对照这一图案,调整电视机的亮度、色彩和对比度,直到满意为止。根据学习机发出的明快而有节奏的"嘟、嘟……"信号声,把电视机的音量调整在适当位置。

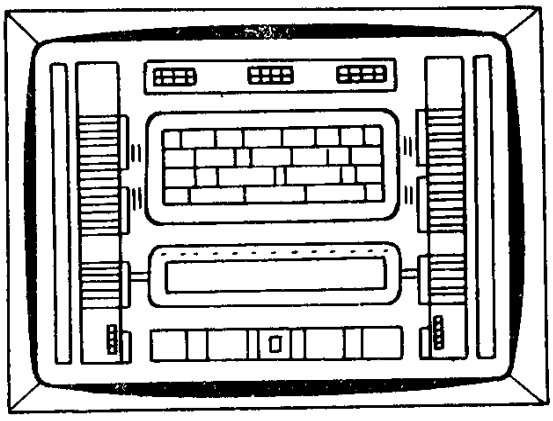


图 1—13 开机画面

5. 工作状态的转换

• 同时按 Ctrl＋Alt＋F2 三个键，系统即可转换至“BASIC”程序设计状态. 同时“BASIC”状态指示灯亮。（注：使用 BS 卡的键盘无此功能）

同时按 Ctrl＋Alt＋F3 三个键，系统即可转换至“GAME”（游戏）状态，同时“GAME”指示灯亮。（分离式结构的电脑只要换游戏卡就可以了）

• 同时按 Ctrl＋Alt＋F1 三个键，系统又可转换至“BS”状态，同时“BS”状态指示灯亮（这一状态是所有键盘都具备的）。

1.5.2 键盘介绍及屏幕编辑方法

1. 键盘介绍：

开机后，同时按 Ctrl＋Alt＋F2 键，系统转换到“BASIC”状态（我们称为直调“BASIC”），同时，“BASIC”状态指示灯亮。在此状态下，可熟悉键盘上各键位的功能。

现在，屏幕上出现下列文字：

```
F BASIC V3.10
C CNEDC SHENZHEN CO.
8182 BYTES FREE
OK
□
```

第一行说明 F BASIC 语言的版本 V 3.10。

第二行是“中国电子器件公司深圳分公司”的缩写。

第三行说明系统有 8182 个单元供用户使用。

第四行“OK”是执行完毕提示符，表示 F BASIC 系统启动完毕。（注：在学习机中，每种操作完毕都要显示“OK”提示符。）

第五行“□”方块表示闪烁的光标，等待用户输入字符。

(1) 4$ ——数字键。（0～9 共 10 个）分上下两档。直接按此键，电脑显示下档数字“4”；若再同时按“Shift”键

则显示上档符号“$”。其余数字键的功能类推。

(2) Esc ——系统退出键。利用该键可使系统从当前功能中退出，还可暂停程序列表。

(3) Tab ——跳格键。每按此键一次，可使光标“□”向右移8格。

(4) Caps Lock ——字母大/小写转换键。按此键后，光标由“□”变成“-”，字母则由大写变成小写，如再按此键则又由“-”变为“□”，恢复字母大写。

(5) Ctrl ——控制键。与其它键同时使用，可完成一系列特定的控制功能。

如：按 Ctrl+G 键，发出蜂鸣声；

按 Ctrl+Alt+Del 键，系统复位并重新启动。

(6) ↑ Shift ——上档转换键。同时按此键与标有上、下两档功能的字符键，可输入字符键上档所标注的符号。同时按此键与字母键，可改变字母的大、小写。

如：按 Shift+“<”键则显示“<”。按 Shift+F 键则显示字母“f”。

(7) Alt ——状态转换键。与其它键同时使用，可完成一系列特定功能的操作。

如：Ctrl+Alt+F1 可进入“BS”状态

Ctrl+Alt+F3 可进入“GAME”状态。

(8) [ ] ——空格键。按此键，在光标位置上会留下一个空格，也可用此键来删除光标所在位置的字符。

(9) N ——字母键(26个)。输入字母用。

(10) < , ——符号键(10个)。用法与数字键相同。

(11) Enter ——输入键(也称回车键)。在通常情况下,输入一个命令后,按此键通知电脑接收,同时光标下跳一行。

(12) Clr Home ——定位/清除键。按此键,光标回到屏幕左上角的开始位置。同时按 Shift+Home 键,可清除整个画面。

(13) Ins —— 插入键。按此键可在光标处插入一个空格。连续按键,可插入多个空格。

(14) Del ——删除键。用来删除光标前面的字符,同时光标后面的所有字符均前进一格。连续按键可连续删除。

(15) ↑ ——方向键(4 个)。可控制光标上、下、左、右移动。连续按键可连续移动。

(16) F1 ~ F8 ——特殊功能键(也称自定义功能键)。用以设置一些特定的功能。

(17) Stop ——停止键。在屏幕编辑时,按此键光标可下降一行;在程序执行中,按此键可中断程序的执行;在程序列表时,按此键可中断列表。

2. 屏幕编辑方法

当我们通过键盘向计算机输入程序命令时,输入的内容也同时显示在屏幕上,若发现错误或格式不对,就要进行修改,这一修改过程就叫做屏幕编辑。

屏幕编辑是借助于电脑中的编辑程序来完成的,它要通过编辑键才能实现。常用的编辑键有"→、←、↑、↓"四个方向键,空格键、"Ins"键、"Del"键、"Home"键和"Shift+Home"键。现介绍使用方法。

(1)擦去多键入的字符

在键入语句过程中，若发现了多键入的字符，可使用“←”键将闪烁光标移到要擦去字符的右边，按“Del”键，就可将该字符擦去。

例如：

已键入：　　10　INPUTT X □

（“□”代表闪烁光标）

发现多键入了一个字符“T”则

按“←”键：　　10 INPUTT□ X

按“Del”键：　　10 INPUT □ X

按“→”键：　　10 INPUT X □

再继续键入以后的字符。

(2)更改已键入的字符

在键入语句过程中，若发现已键入的字符是错误的，需要更改时，可使用方向键将光标移到要更改的字符上面，直接键入正确的字符即可。

例如

已键入：　　30 PRINN “ABSD”□

发现：“PRINT”中的第五个字符误键入“N”，则按“←”键，使

30　PRINT [N]“ABSD”

键入“T”：　　30 PRINT□ “ABSD”

键“→”：　　30 PRINT “ABSD” □

(3)增添漏键入的字符

在键入语句过程中，若发现漏键入了字符，可使用方向键把光标移到欲添加字符处，按“Ins”键，则出现一个空格(即光标后的所有字符自动后移)，在空格处键入欲添加的字符。

例如

已键入：　　20 PRINT“FCS－90 MICROCOMTER”□

在微型电子计算机一词“MICROCOMPUTER”中漏掉了“PU”两个字符。

按“←”键： 20 PRINT“FCS－90 MICROCOM T ER”把光标移到“PU”的加入位置“M”的右边。

按“Ins”键两次，则有

20 PRINT “FCS－90 MICROCOM□ TER”

键入“PU”：20 PRINT“FCS－90MICROCOMPU T ER”

按“→”键：20 PRINT“FCS－90MICROCOMPUTER”□

(4)擦除语句前面的一串字符

在编写程序时，键入一行语句后，发现前边的部分字符是多余的，需要擦除时，可使用“←”键，把光标移到待擦除字符的右边，连续按“Del”，擦掉所有要擦除的字符。

例如：

已键入： 40 PRINT“X＋Y”:INPUT X□

现在要擦去 PRINT“X＋Y”:11 个字符，可按“←”键，把光标移到“1”上，

40 PRINT“X＋Y”：I NPUT X

按“Del”键 11 次，则有

40 I NPUT X

删去了前面的 11 个字符，且把“INPUT X”移到了语句开始位置。

(5) 擦除语句后面的一串字符

上例中如欲擦去“:INPUTX”七个字符，可连续按“Del”键八次，也可以把光标移到“:”处，按住空格键，直到字符全部擦除后再松开。

(6) 擦除整条语句

当发现已键入的语句无用，或因键入语句的错误太多，欲全部擦除，再重新键入新的语句时，可同时按“Shift＋Home”两键，即可把全屏幕清除干净，且光标回到屏幕左上角开始位置。

例如屏幕上现有内容为

10 PRINT "XYZ":INPUT"X="X:

INPUT "y="Y□

同时按"Shift+Home"则只剩下闪烁的光标:□

(7) 光标的快速移动

按住四个方向键不放松,稍停一会,光标即会连续移动。

按"Home"键,可使处于任何位置的光标跳到屏幕左上角。

按住"Del"不放松可连续擦除。

按住"Ins 不放松可连续产生空格。

总之屏幕编辑方法灵活,技巧性较强,只有加强练习,才能熟练掌握,使用得心应手,才能提高屏幕编辑速度和编程工作效率。以后的学习中还会看到,使用"Ctrl+E"键可擦去光标所在行的全部语句;"Ctrl+L 键可清除整个屏幕,相当于"Shift+Home";"Ctrl+Z"键可以清除光标以后的所有字符等屏幕编辑操作。

1.5.3 电脑游戏机的功能选择

在"BS"(学习)状态下,学习机可以通过"人机对话"方式进行F BASIC 程序设计、键控绘图、计算、谱曲与演奏和打字训练等多种功能的选择。

1."人机对话"画面

开机后,系统进入"BS"状态,屏幕显示下述"人机对话"彩色画面。

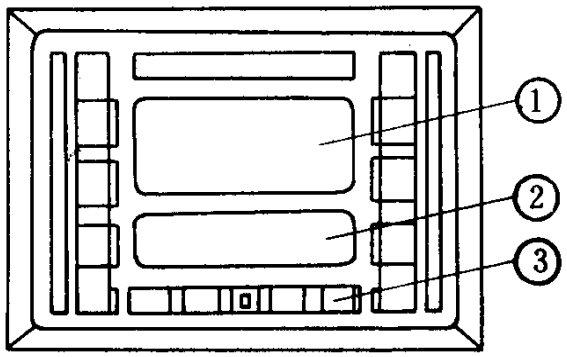


图 1—14 人机对话画面

①区：显示电脑的对话内容。（现在该区显示的是彩色棋盘方格图案）

②区：显示你的对话内容。

③区：显示操作提示。

2. 操作方法

(1)按“Enter”键(或空格键)，在①区将依次显示如下内容：

```
I'M CNEDC FAMILY
COMPUTER SYSTEM.
NOW WE START.
```

“我是中国电子器件公司(CNED)家用电脑系统。现在我们开始对话。

```
WHO ARE YOU?
INPUT NAME.
```

“我怎么称呼你？请输入你的姓名。”

(2)当②区出现闪烁光标时，用英文或汉语拼音输入你的姓名。假定你叫李华，则键入“LI HUA”，此时②区显示如下内容：

```
□ LI HUA
```

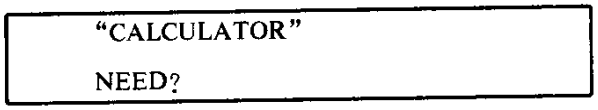
(3)按“Enter”键，系统确认你的姓名后，在①区依次显示如下内容：

```
OK!
LI HUA
IS YOU.
NOW WE START.
```

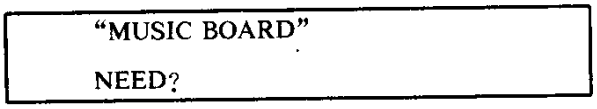
“好！你是李华。现在我们开始选择功能。”

接着屏幕依次循环地显示下述功能选择询问：(每次只显示一个询问)

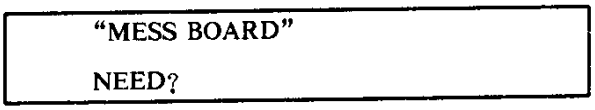
"你要选择 F BASIC 程序设计功能吗?"



"你要选择计算器功能吗?"



"你要选择谱曲和演奏功能吗?"



"你要选择打字练习功能吗?"

在显示上述四项功能选择询问之一的同时在③区显示如下操作提示：

F1	F2	F3	F4
OK	NO	HELLO	END

(4)选择：

• 如你想选择某个功能，只要在该功能选择询问出现时，按"F1"键(或键入"OK")，再按"Enter"键，系统就会进入该功能。

• 如你不愿选择当前询问的功能，可按"F2"键(或键入：NO")，再按 Ener"，屏幕就会显示下一个功能选择询问。

• 按"F3"键(或输入："HELLO")，再按"Enter"键，系统便会通

过屏幕向你问候致意。

• 按“F4”键(或输入“END”),再按“Enter”键,则系统又回到“人机对话”画面。

(5)利用指令功能

不论系统显示何种功能选择询句,都可用下述指令选择所需的功能:

• 键入“BASIC”,再按“Enter”键,系统进入 F BASIC 程序设计功能状态。

• 键入“CAL.”,再按“Enter”键,系统进入计算器功能状态。

• 键入“Mus.”,再按“Entrep”键,系统进入谱曲和演奏功能状态。

• 键入“MES.”再按“Enter”键,系统进入打字练习功能状态。

• 键入“END”(或按 F4 键),再按“Enter”键,系统又回到“人机对话”画面。

(6)各项功能选择操作流程图

为了方便读者进行各项功能选择操作,请参阅图 1—15 所示的功能选择操作流程图。

1.5.4 算术运算(CALCULATOR BOARD)

本功能可供小学生练习算术四则运算。

1. 功能及说明

(1)可进行“+”、“—”、“×”、“÷”及四则混合运算。

(2)运算最多允许有 8 位有效数字,其范围为±(0.0000001～99999999)

• 计算式中如出现 8 位以上的有效数字时,系统将拒绝运算,并给出“WRONG”(错误)信息。

• 计算结果中,如整数部分多于 8 位,系统将显示“OVER-FLOW”(溢出)信息。

• 计算结果中,当有效数字多于 8 位,但整数部分不多于 8 位

时，则系统自动删除 8 位以后的小数。

(3)算式中可以使用多重括号，括号无大、中、小之分，一律使用圆括号“()”。计算式中最多允许使用 10 个括号。

(4)系统进行运算时，遵循四则运算法则，即先“()”，后“×”、“÷”，最后“+”、“−”。

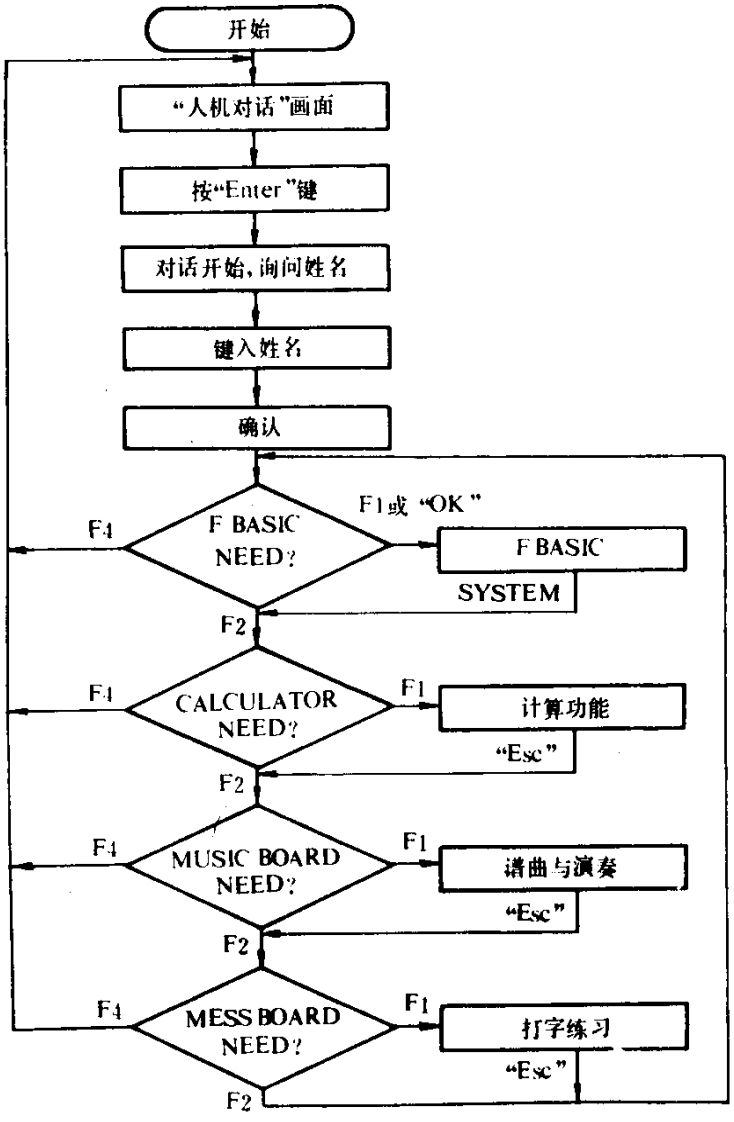


图 1−15　功能选择操作流程图

2.　运算操作

(1)在“BS”状态下，利用“人机对话”画面，当系统询问“CALCULATOR NEED?”时，键入“F1”或“OK”则系统进入 CALCULA-

TOR 功能状态，此时，屏幕显示如下画面（见图 1－16）。

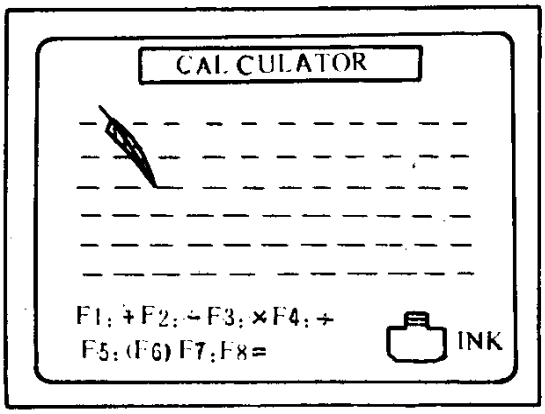


图 1－16　算术运算画面

（2）使用数字键和 F1～F8 键书写（输入）计算式，F1～F8 各键的意义如下表所示：

按　　键	F1	F2	F3	F4	F5	F6	F7	F8
对应的算符	＋	－	×	÷	(	)	·	＝

（3）“羽毛笔”指示当前书写计算式的位置，可用四个方向键移动羽毛笔到任意位置。

（4）整个屏幕有 8 行空间供书写算式。每行最多可输入 24 个字符（包括空格）。

（5）同一行可输入多个算式，但两个算式之间至少要留有一个空格，否则系统会显示“WRONG”信息。

（6）如计算式太长，在一行中书写不下，可占用下一行继续书写。

（7）算式中不允许有空格，否则空格前的内容不能计算。

（8）在计算结果后面允许输入字符，也允许在字符之后继续输入计算式，但字符与计算式之间至少要有一个空格。

（9）计算式中不允许输入字符，否则将显示“WRONG”信息提示。

(10)运算符后面的负数要加括号。如：

$$-3+-5$$

应书写为

$$-3+(-5)$$

(11)8行空间写满后，可按“Home＋Shift”键清除屏幕显示的内容，从第一行开始书写。

3. 退出计算器功能的操作

系统处于计算器功能状态时，按“Esc”键，则结束计算器功能，又回到功能选择询问状态，屏幕将依次显示如下内容：

“CALCULATOR”
NOW FINISHED.

“运算功能现在已结束。”

“NEXT SELECT”.

“进行下一个功能选择”

“MUSIC BOARD”
NEED?

“你是否要选择谱曲与演奏功能？”

若不要其它功能，则按“F4”键，再按“Enter”键，系统又回到“人机对话”画面(图1－13)。

4. 操作出错解释

如出现下列情况之一，系统会显示“WRONG”或“OVERFLOW”提示(即“错误”或“溢出”)。

(1)计算结果中，整数部分多于8位(图1－17)；

(2)计算式中的括号多于10个或出现双括号(图1－17 b)；

(3)两计算式之间没有空格(图1－17 c)；

(4)字符与后面的算式之间没有空格(图 1－17 d)；

(5)计算式中出现除数字和运算符以外的字符(图 1－17 e)；

(6)等号的左方是空格(图 1－17 f)；

(7)运算符后面的负数没有加括号(图 1－17 g)

88888888×138=OVERFLOW	a
(3-6)+8×((7-5)×6)=WRONG	b
4+3-1=67+5=WRONG	c
3+5=8 OK 6+5=WRONG	d
8YUAN×3=WRONG	e
678×3+13=WRONG	f
－6+－4=WRONG	g

图 1－17　操作错误示例

5.　运算功能的应用

使用运算功能，可对小学生算术四则运算进行验算。由孩子信自己操作电脑验算作业，不仅节省了家长的辅导时间，而且还培养了孩子们学习电脑的兴趣和爱好。

使用中，可以书写一道题，运算一个答案；也可以先写入一系列算式，自己先求出答案，最后再按“＝”键(即 F8)逐一调出答案，与自己的计算结果进行核对。

1.5.5　谱曲与演奏

本功能可供读者进行谱曲练习，并能把谱写在屏幕上的乐曲自动地演奏出来。

1.　使用说明

(1)　屏幕的划分

谱曲屏幕分为四行，每行分上、中、下三段，供谱曲和配“和音”用。每行可输入 24 个音符(包括空格)，屏幕下部显示有操作提示，如图 1－18 所示：

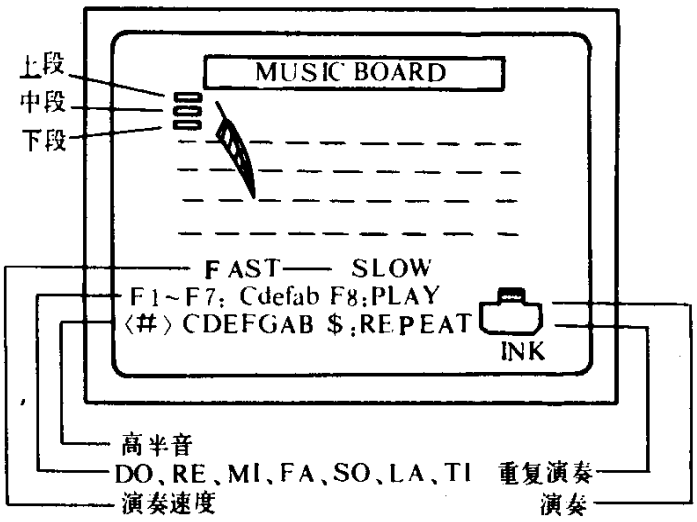


图 1—18　谱曲画面

(2)按键与音符的对应关系

← “F1～F7”键分别代表 c、d、e、f、g、a、b 七个音名，也可用小写字母键“c、d、e、f、g、a、b”输入，它们分别表示简谱中的“1、2、3、4、5、6、7”(即 DO、RE、MI、FA、SO、LA、TI)。

• Shift＋“F1～F7”键分别表示 C、D、E、F、G、A、B，也可以用大写字母键“C、D、E、F、G、A、B”输入，它们表示七个音阶的高半音，即高半音的 1、2、3、4、5、6、7。

• 每行乐曲的上段为高音域。在上段输入“c、d、e、f、g、a、b”，就会发出 $\dot{1}$、$\dot{2}$、$\dot{3}$、$\dot{4}$、$\dot{5}$、$\dot{6}$、$\dot{7}$的乐音；如果输入数字键的 1、2、3、4、5、6、7 就会发出中音域的 1、2、3、4、5、6、7 的乐音。

• 每行的中段为中音域。在中段输入 c、d、e、f、g、a、b，就会发出中音域的 1、2、3、4、5、6、7 的乐音；如果输入数字键的 1、2、3、4、5、6、7 就会发出高音域的 $\dot{1}$、$\dot{2}$、$\dot{3}$、$\dot{4}$、$\dot{5}$、$\dot{6}$、$\dot{7}$的乐音。

• 每行的下段为低音域。在下段输入 c、d、e、f、g、a、b 就会发出低音域的 $\underset{\cdot}{1}$、$\underset{\cdot}{2}$、$\underset{\cdot}{3}$、$\underset{\cdot}{4}$、$\underset{\cdot}{5}$、$\underset{\cdot}{6}$、$\underset{\cdot}{7}$的乐音；如输入数字键的“1～7”则会发

出更低的低音 $\dot{1}$、$\dot{2}$、$\dot{3}$、$\dot{4}$、$\dot{5}$、$\dot{6}$、$\dot{7}$的乐音。

在三段音域中字符与音符的对照见表 1—5。

表 1—5　字符与音符对照表

音段 / 字符别	上段	中段	下段
F1～F7 键	$\dot{1}\sim\dot{7}$	$\dot{1}\sim\dot{7}$	$\underset{\cdot}{1}\sim\underset{\cdot}{7}$
1～7 数字键	$1\sim7$	$\dot{1}\sim\dot{7}$	$\underset{:}{1}\sim\underset{:}{7}$
CDEFGAB	$\dot{1}^{\#}\sim\dot{7}^{\#}$	$1^{\#}\sim7^{\#}$	$\underset{\cdot}{1^{\#}}\sim\underset{\cdot}{7^{\#}}$

2.　谱曲操作

(1)“羽毛笔”指示当前输入音符的位置。按方向键可以移动“羽毛笔”到欲输入音符的位置。

(2)输入音符：

·　在上段如要输入音符 $\dot{1}$、$\dot{2}$、$\dot{3}$、$\dot{4}$、$\dot{5}$、$\dot{6}$、$\dot{7}$，按“F1～F7”键(或按小写字母键 c、d、e、f、g、a、b)，在屏幕上相应地显示 c、d、e、f、g、a、b。

·　在上段如要输入高半音的 $\dot{1}\sim\dot{7}$，按 Shift＋“F1～F7”键(或按大写字母键 C、D、E、F、G、A、B)，在屏幕上相应地显示 C、D、E、F、G、A、B。

·　在上段如要输入中音域的 1～7，按数字键 1～7，在屏幕上相应地显示 1～7。

·　在中段如要输入中音域的 1～7，按 F1～F7 键(或按小写字母键 c、d、e、f、g、a、b)，在屏幕上相应地显示 c、d、e、f、g、a、b。

·　在中段如要输入高半音的 1～7，按 Shift＋“F1～F7”键(或按大写字母键 C、D、E、F、G、A、B)，在屏幕上相应地显示 C、D、E、F、G、A、B。

• 在中段如要输入高音域的 1～7，按数字键 $\dot{1}$～$\dot{7}$，在屏幕上相应地显示 1～7。

• 在下段如要输入低音域的 $\underset{\cdot}{1}$～$\underset{\cdot}{7}$，按 F1～F7 键(或按小写字母 c、d、e、f、g、a、b)，在屏幕上相应地显示 c、d、e、f、g、a、b。

• 在下段如要输入比低音低 8 度的 $\underset{\cdot}{1}$～$\underset{\cdot}{7}$，按数字键 1～7，在屏幕上相应地显示 1～7。

(3)重复演奏

如要使乐曲重复演奏，可在乐曲的末尾使用重复演奏符“$”，输入办法是同时按 Shift＋4(不论上、中、下段均可)，这时屏幕上显示“$”。

(4)乐谱的修改

• 使用方向键移动“羽毛笔”到欲修改的音符位置，输入正确的音符。

• 当系统处于演奏状态(不论是正在演奏或已中止演奏)，“羽毛笔”都是插在墨水瓶中。此时，如欲修改乐谱，可按 Home 键将“羽毛笔”移到屏幕的左上方，再使用方向键来修改。

• 按“Del”键可删除“羽毛笔”左方的一个音符(每按一次，删掉一个)。

• 按“Ins”键可在“羽毛笔”左方插入一个空格(每按一次，插入一个)。当需要插入字符时，可先移出空格，再加入字符。

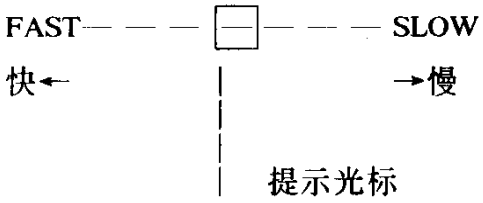
(5)乐谱的清除

当欲清除屏幕上的全部乐谱时，同时按 Shift＋Home 键可清除屏幕上显示的全部字符。

3. 乐谱的演奏

(1)按 F8 键，系统就会按照屏幕上的乐谱自动演奏。(上、中、下三段音符同时奏出。)在演奏中途按 F8 键，系统会从头开始重新演奏。

(2)使用“←、→”键可改变演奏中的演奏速度，并在屏幕下部提示区用下图提示：



(3)中止演奏

按空格键可中止演奏。此时方格光标便停留在中止演奏的那个音符上。按“←”或“→”键可以逐个音符向右或向左弹奏。

4. 谱曲和演奏功能的退出

按“Esc”键，系统退出谱曲、演奏功能，回到功能询问。

5. 示例

为便于读者练习，特缩编了《铃儿响叮当》三重奏曲，读者只要按照下面的乐谱，输入系统后，按F8键，就可反复欣赏这首世界名曲。

JINGLE BELL(铃儿响叮当)

```
上 eeeeeeegcde   ffffeeeddde
中 c cg ec e g e c fg ed a
下 c gc gc g ccdef ac gd
.....
上 dgeeeeeeeegcde     ffffeeg
中 gbc eg ec e g e c fg ed
下 dgfc gc gc g c cdef ac gd
.....
上 gfdc  5edc5  5edc6  6
中  f c  c e g e c e f d a
下 fg5c567c g c555c g f666 f
.....
上 fedgggagfdc
中  f b g f d c $
下  a g b g567c5
.....
```

1.5.6 打字练习/留言

本系统的键盘按国际标准键盘设计，可进行英文打字练习。

1. 功能说明：

本系统具有三项功能：

(1)进行英文打字练习。

(2)利用本系统特备的图形，制作表格。

(3)利用屏幕作为留言板。

2. 操作说明：

(1)当系统选择功能询问至

"MESS BOARD"
NEED?

按"F1"键或键入"OK"，则系统进入打字练习功能状态。屏幕显示如下：

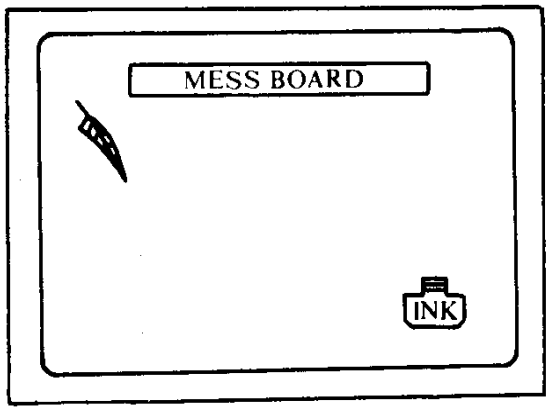


图 1—19 打字画面

(2)使用键盘进行英文打字练习，同时屏幕显示键入的内容。(建议一开始就按照英文打字教程规定的指法进行练习，以养成良好的习惯。)

(3)利用"Caps Lock"键进行大/小写字母的转换。(大写状态，"羽毛笔"为红色，小写状态"羽毛笔"为绿色。)

(4)当处于小写状态时("羽毛笔"为绿色),使用"Alt"键和数字键,可得到一系列制表用的图形,如表 1—6 所示

表 1—6 制表图形与按键对照表

Alt +	1	2	3	4	5	6	7	8	9	0	—	=	¥
对应的图案	┼	┴	┬	┤	├	─	│	┌	┐	└	┘	□	▨

注: "Alt"+"="为白色背景方块

"Alt"+"¥"为红色背景方块

(5)按"Enter"键,"羽毛笔"下移一行。

(6)全屏幕显示 24×17 个字符,(即显示 24 列,17 行)

(7)当整个屏幕写满后,"羽毛笔"会自动移到第一行起始位置。如果再输入字符,新输入的字符就会把原来输入的字符复盖掉。

(8)修改

- 使用方向键移动"羽毛笔"到要修改的字符处,然后键入正确的字符。
- 按"Del"键,可删除"羽毛笔"左边的一个字符。
- 按"Ins"键,可在"羽毛笔"左边插入一个空格。

(8)清屏

在大写状态下("羽毛笔"为红色),按"Shift+Home",可清除屏幕显示的内容。

3. 退出打字、留言功能状态

按"Esc"键,系统退出打字、留言功能,回到功能选择询问。

1.5.7 程序设计

本系统设有 F BASIC 程序设计功能,可使用 F BASIC 语言进行程序设计。

1. 如何进入 F BASIC 功能状态

本系统有两种方式进入 F BASIC 功能状态：

(1)通过“功能选择”进入 F BASIC

系统在“BS”状态，通过“人机对话”画面，将显示一系列功能选择询问。当询问到：

```
F BASIC
NEED?
```

时，按 F1 键(或输入“OK”)，再按“Enter”键，或者在任一功能选择询问下，键入“BASIC”再按“Enter”键，都可进入 BASIC 功能状态。这时屏幕上显示 F BASIC 选择菜单(见图 1—20)其中：1 为 F BASIC 程序设计；2 为利用背景键控绘图；3 为返回“人机对话”画面。可根据需要进行选择。

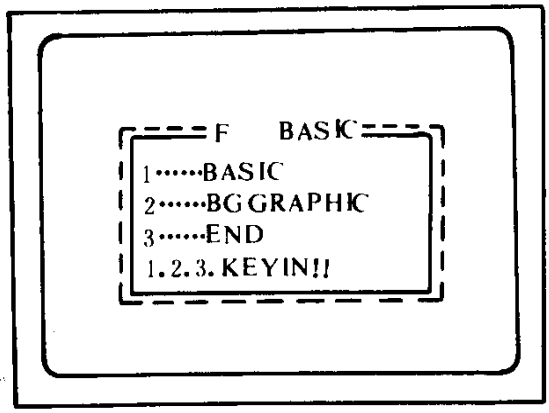


图 1—20 F BASIC 菜单

如果要进行 BASIC 程序设计，则按数字键“1”，屏幕将显示图 1—21 的画面。“OK”及闪烁光标“□”告诉你可以进行程序设计了。这时系统有 4030 个内存单元供用户使用。我们把这种进入 F BASIC 的方式称为“对话进入 BASIC”(即 BS. 2A F BASIC)。

注：如要返回 F BASIC 选择菜单，可键入“SYSTEM”，再按“Enter”键。

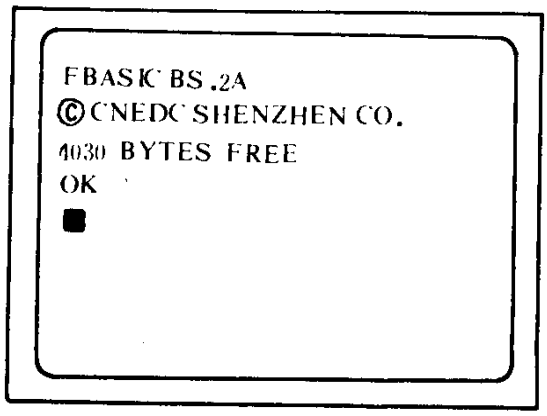


图 1—21　对话 BASIC 画面

• 如果要进行 BG GRAPHIC(利用背景图形绘图，简称“键控绘图”)则按数字键“2”，(关于绘图操作将在下一节介绍)即可进入 BG GRAPHIC 状态。

注：如果从 BG GRAPHIC 回到 F BASIC 选择菜单，先按“Esc”键，再按“Stop”键。

• 如按数字键“3”则系统由 F BASIC 选择菜单回到图 1－14“人机对话”画面。

2. 直接进入 F BASIC

不论系统在何种状态，只要同时按“Ctrl＋Alt＋F2”键，立即进入 F BASIC 程序设计状态，同时“BASIC”指示灯亮，屏幕显示：

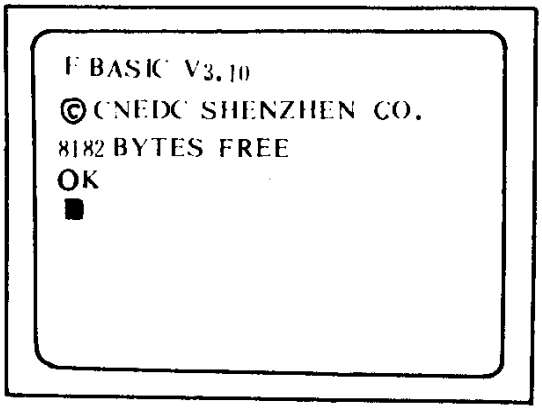


图 1—22　直调 BASIC 画面

我们把这种进入F BASIC的方式，称为直调“BASIC”。(即V3.10 F BASIC)

以后的介绍中将会看到，直调BASIC系统将比对话BASIC系统提供更多的用户内存空间和更完善的指令，供用户方便地设计比较复杂的应用程序。

更有趣的是，直调BASIC系统中固化有四个不完善的游戏程序，按F1～F4键可从系统中调出。例如：按F1键，屏幕上将显示出游戏画面。可以发现，这个游戏不能正常进行。只有把程序补充、修改完善后才能运行。待读者学习完本书的上篇，基本掌握F BASIC的指令后，即可较容易地完成上述的补充、修改工作。所以，这四个小程序可以说是对读者掌握F BASIC语言熟练程度的测试。

如果要使系统从F BASIC状态中退出，回到“BS”状态，只要同时按Ctrl＋Alt＋F1键即可返回图1－14的画面。

3. BASIC功能选择操作流程

为方便读者操作，特画出“进入或退出F BASIC功能操作流程图”(图1－23)和“从任何状态进入或退出BASIC状态的操作流程图”(图1－24)。

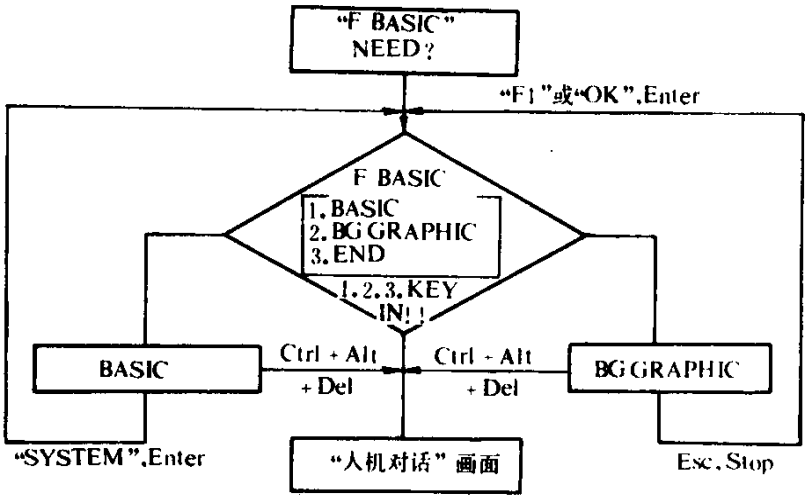


图1－23 进入或退出对话BASIC功能操作流程图

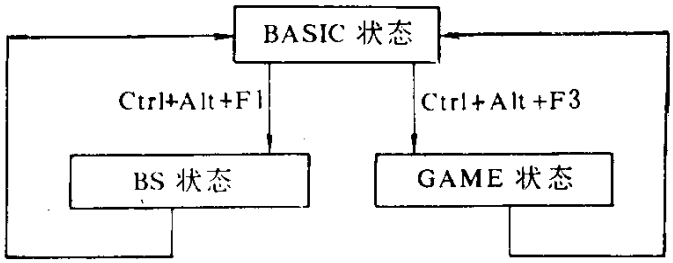


图 1—24　进入或退出直调 BASIC 状态操作流程图

1.5.8　键控绘图(BG GRAPHIC)

BG GRAPHIC(Background Graphic)是利用背景图形绘画的意思,是指利用系统内部储存的具有 4 种颜色变化的 104 种图形来作图。绘图方法简便,图案妙趣横生。

1.　绘图系统介绍

开机后,经"人机对话",选择"F BASIC"功能状态,屏幕显示如图 1—20 所示的 F BASIC 选择菜单,按数字键"2"后,系统进入 BG GRAPHIC　键控绘图状态,屏幕显示如图 1—25 所示的画面。

画面说明如下:

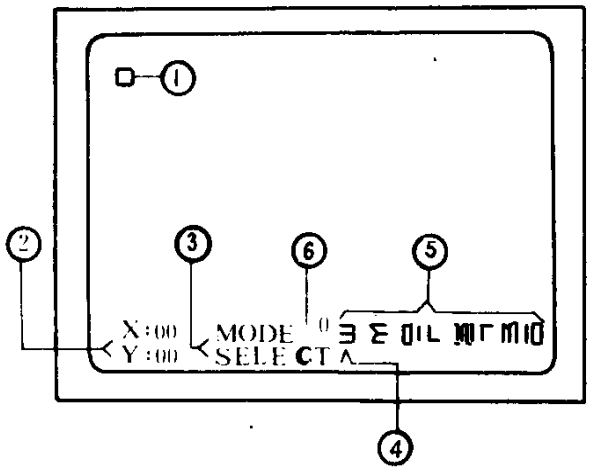


图 1—25　键控绘图画面

(1)光标

光标指示作图位置。操作时使用方向键使光标移到设定的位置。

(2)光标的坐标值

当光标在屏幕上移动时,其坐标值能自动跟踪并显示:

显示范围:横坐标 X:00～27

纵坐标 Y:00～20

(3)操作指令状态提示

键控绘图共有六条指令可供选择。一旦系统进入某条指令状态,就在该位置显示某条指令。

(4)图形选择指示符

对于屏幕右下角显示的 8 个图形,可用指示符"^ "的移动,来确定绘图所需的图形。

按"Ins"键,指示符"^ "向左移动;按"Del"键,指示键"^ "向右移动。

(5)图形

供绘画用的背景图形共 104 个,分为 13 组,每组 8 个图形,每次显示一组。

连续按"Home"键,图形从第 1 组到第 13 组依次显示。按"Shift+Home"键,图形从第 13 组到第 1 组依次显示。(每按一次键,显示 1 组图形),图形种类见附录。

(6)配色号码

图形颜色分为 4 种,颜色编号分别为 0～3。按"Enter"键,图形颜色改变,同时显示相应的颜色编号(通称:配色号码)。

2. 绘图指令介绍

按"Esc"键,屏幕左上角显示绘图操作指令菜单,如图 1－26 所示。

使用指令选择符">"可选择所需的操作指令。利用"↑"、"↓"键调整">"的位置。选择确定后按空格键,系统进入所选定的操作指令状态。

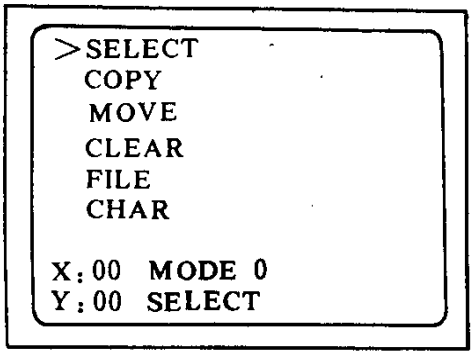


图 1—26　绘图指令选择画面

(1)SELECT——选择背景图形绘画指令

系统进入键控绘图功能状态后，自动设定为“SELECT”状态。在该状态下可自由选择背景图形和颜色绘出各种图画。若在其它指令状态下，只要按“Esc”键，便显示图 1－26 的绘图指令选择画面。此时，选择符“>”处于 SELECT 指令位置，按空格或“Enter”键，系统即可进入 SELECT 指令状态。在该指令状态，绘图方法如下：

① 按“Home”或“Shift＋Home”键，屏幕右下角分组显示各种图形，以供作图选择。

② 按“Del”或“Ins”键，使图形指示符“ ^ ”移到选定的图形位置。

③ 按“Enter”键，选定图形的配色。

④ 使用方向键，使光标“□”移到所需绘画的位置。

⑤ 按空格键，所选定的图形便在光标“□”处显示。此时，光标“□”自动向右移动 1 格。若要使该图形连续显示，可连续按空格键。

⑥ 绘画的修改：按“D”键，可使光标“□”处的图形删除。如欲修改某处的图形，可把光标移到欲修改处，按空格键，即可用新图形把旧图形复盖。

⑦ 示例：按照上述操作步骤画出如图 1－27 所示的机器人。

绘画步骤：

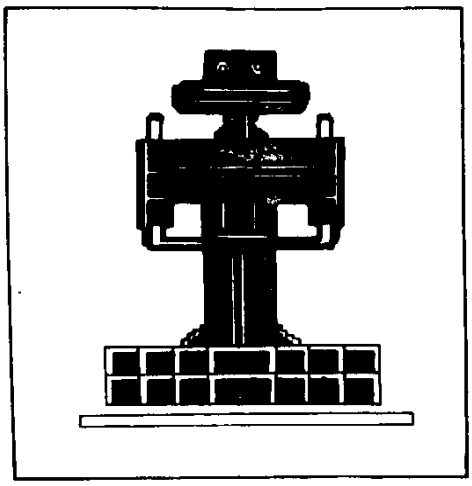


图 1－27 机器人

从第四组图形中选择第五个图形画眼睛。

从第 6 组中选择第五、六、七图形画脸。

选第 4 组中第二个图形画左边脖子。

选第 1 组中第四个图形画右边脖子。

选第 2 组中第七、八图形画肩、胸。

选第 1 组中第五个图形画左边手臂。

选第 4 组中第三个图形画右边手臂。

选第 3 组中第五个图形，选第二组中第 1 个图形画上腹部。

选第 1 组中的第一、二图形画下腹部。

选第 1 组中第三个图形画左腿。

选第 3 组中第七个图形画右腿。

选第 2 组中第二个图形画左脚。

选第 3 组中第二个图形画右脚。

选第 3 组中第七个图形画左边台阶。

选第 4 组中第八个图形画右边台阶。

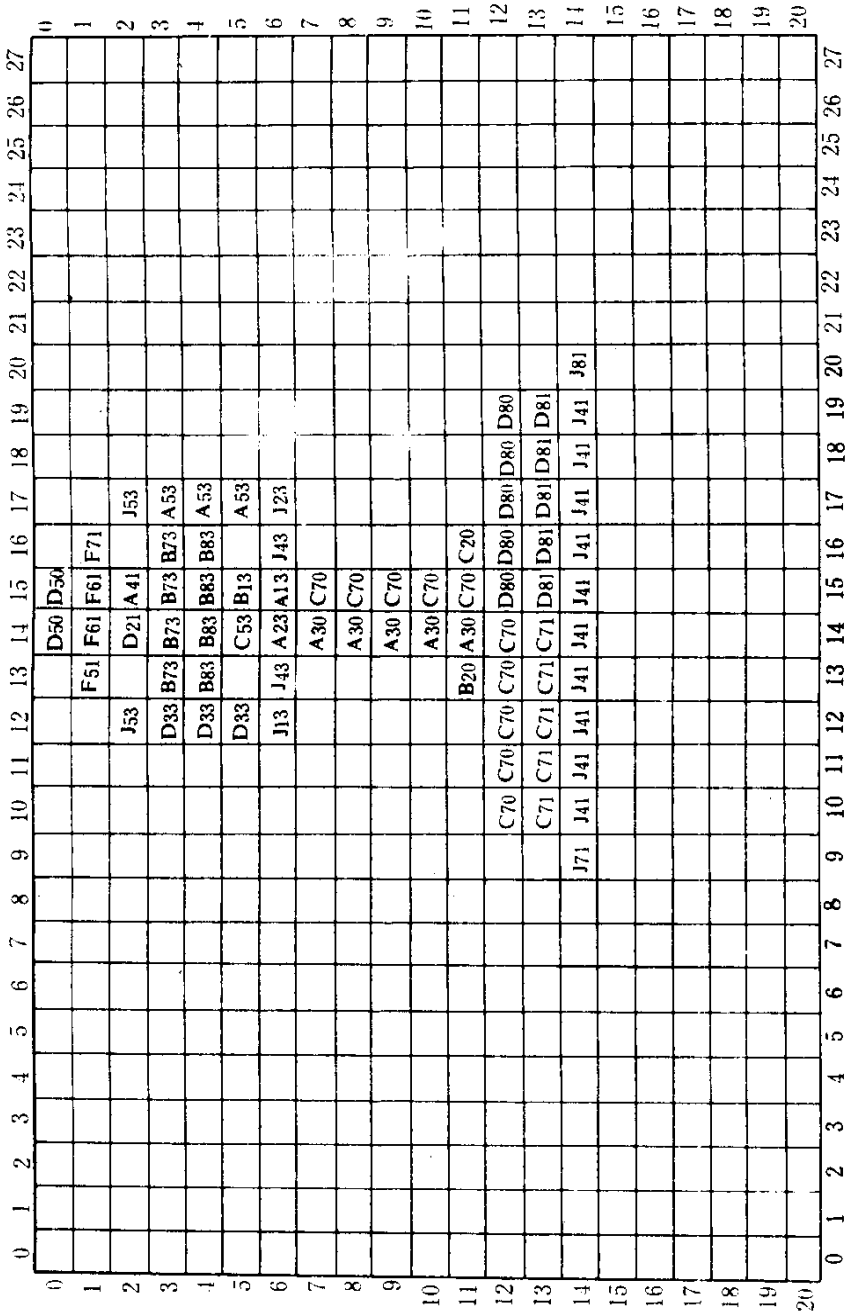


图 1—28　机器人绘图规则

选第 10 组中第一、四、二、五、七、八图形画连接线及底线。

对照图 1－27 修改错误即得到一个逗人的机器人的图案。

读者可能觉得上述绘图过程的叙述较繁琐，而且不直观。每一个图形的定位不明确，开始绘图时，要试好多次才能安放恰当。因此，绘图步骤的叙述中还应加入每个图形的座标。另外，每个图形还有四种色彩的变化，所以，绘图叙述中还应加入每个图形的颜色代码。可想而知，这样的绘图叙述要更加复杂。

为使键控绘图过程的叙述简单明了，今后我们采取图 1－28 的形式说明。图中每一组字符串表示一个固定颜色的图形。如："B83"表示背景图形库(封二)中第 B 行第 8 个图形，颜色代码为 3。每个图形的纵、横座标可从图中直接读出，对照光标的提示座标即可方便地定位。

(2)COPY—复制图形指令

COPY 指令可对画面中已有的图形在其它位置上进行复制，同时该图形仍保留在原位置。从而省略了重新到 104 个图形中去选择的操作过程，为在一个画面中绘制多个相同的图案提供了简便的方法。

操作步骤：

① 按"Esc"键，使用"↓"键从显示的指令群中选择 COPY 指令，然后按空格键，系统立即进入 COPY 状态，同时在图 1－23 中的"操作指令状态提示"一栏显示："COPY"。

② 使用方向键控制光标"□"移到需复制图形的位置上。

③ 按一下"Ins"键，确认需复制的图形。(此时该处的图形便被记忆在光标中)

④ 使用方向键使光标"□"和被确认的图形一起移到设定的位置。

⑤ 按"Del"键，即完成一个图形的复制。

⑥ 使用方向键和"Del"键，可使同一图形在不同的位置上进行

复制。

⑦ 按“Ins”键，可解除随光标一起移动的图形记忆，并删掉记忆的图形。

⑧ 图形的擦除：在 COPY 指令状态下，欲擦除画面上的某一个图形，可先确定光标“□”的位置，然后按“Del”键，光标“□”中的图形即被删除。

⑨ 示例：按上述操作步骤，复制一个机器人。

(3)MOVE 图形位置移动指令

MOVE 指令可将画面上的图形移到别处，同时原位置上的图形消失。

操作步骤：

① 按“Esc”键，恢复指令选择画面图 1－26 使用方向键选择 MOVE 指令，按空格键，系统进入 MOVE 图形位置移动状态。

② 使用方向键，将光标“□”移到需要移动的图形位置上。

③ 按“Ins”键，确认需要移动的图形。

④ 使用方向键，使光标和确认的图形一起移到所设定的位置上，同时原位置上的图形消失。

⑤ 按：“Del”键，即完成该图形的移动。此后移动光标，光标内的图形自动消失。

⑥ 图形的删除：

在 MOVE 指令状态下，要删除画面上某一图形，可先确定光标“□”的位置，然后按两次“Ins”键，光标“□”中的图形即被删除。

(4)CLEAR——屏幕图案清除指令

CLEAR 指令，可用来清除绘制在屏幕上的所有图形。一般仅在重新绘图时使用。

操作步骤

① 按“Esc”键，返回图 1－24 指令选择画面，选择 CLEAR 指令，按空格键，进入 CLEAR 状态。

② 按空格键，屏幕上所有的图形立即被清除。清除后，系统自动进入 SELECT——选择背景图形绘画状态。

(5)CHAR——字符打印指令

CHAR 指令状态下，可使用键盘把字母、数字及符号打印在屏幕上。

操作步骤：

① 按“Esc”键. 选择 CHAR 指令，然后按空格键，系统进入 CHAR——字符打印状态。

② 使用方向键移动光标“□”至需打印字符的位置上。

③ 直接通过键盘上的按键输入所需要的字符。(注——字符若打印在图形上，则该处的图形会消失。)

(6)FILE——画面写/读指令

FILE 指令可将绘制的画面写入录音磁带中保存，或者从录音磁带中读出已写入的画面，使该画面在屏幕上显示。

操作步骤：

① 按“Esc”键，选择 FILE 指令，按空格键系统便进入 FILE 指令状态。

② 这时屏幕上显示

“SAVE(S),LOAD(L)? □”

询问是写(SAVE)还是读(LOAD)，若欲写入磁带则按“S”键；若欲读磁带则按“L”键，这时屏幕上第二行将显示

“FILE NAME”□

键入文件名，按“Enter”键后，系统即进行写或读操作。

③ 具体操作见 1.5.10“数据资料的写/读”一节。

1.5.9 电视游戏

1. 系统连接

(1)在主机断开电源的情况下按照图 1—10 所示连接好电路(不要接录音机)。

(2)插入所选用的游戏卡,要保证游戏卡与插座的导向一致,接触良好。

2. 操作步骤:

(1)按游戏学习机的电源键"POWER","BS"功能指示灯亮。

(2)打开电视机电源,使屏幕上显示清晰的"人机对话"画面(图 1—14)。

(3)同时按"Ctrl+Alt+F3"键,"GAME"功能指示灯亮,系统进入游戏状态,并显示游戏目录或图案。

(4)对于分立式的游戏学习机,只要使键盘与游戏机脱离就可以返还游戏机功能。

1.5.10 数据资料的写/读

切断游戏学习机的电源会导致内存的数据资料丢失。因此有必要利用计算机外部存储设备把有用的程序和图画保存(SAVE)起来,以备以后重新取出(LOAD)使用。游戏学习机系统备有连接家用盒式录音机的接口,可供用户向磁带中保存(写入)或从磁带中取出(读出)数据资料。

1. 录音机和主机的连接操作步骤

(1)将一根数据信号传输线的一端插入学习机的"SAVE"插座,另一端插入录音机的"MIC"话筒插座。

(2)将另一根数据信号传输线的一端插入学习机的"LOAD"插座,另一端插入录音机的"EAR"耳机插座。

(3)确认以上连接无误后,接通录音机的电源。

(4)调整录音机的音量和音质至适当位置,一般将音量调到中间位置;音质调至高音位置。

2. 将数据资料保存(写入)录音磁带中

程序的数据编辑完成后,可以进行存储操作:

(1)存储用 BASIC 语言设计的程序,必须使系统处于"BASIC"工作状态。

• 按录音机放音(PLAY)键和录音(RECORD)键。

• 由键盘输入指令.SAVE“文件名”,按“Enter”键即可存储数据资料。

(2)存储用 BG GRAPHIC 绘制的图案必须使系统处于 BG GRAPHIC 工作状态。

• 按“Esc”键,从显示的操作指令群中选择 FILE 指令。

• 按空格键,系统进入保存(写入)/取出(读出)状态。

• 按录音机的放音键(PLAY)和录音键(RECORD)。

• 屏幕显示

SAVE(S),LOAD(L)? □

此时,键入“S”,则在第二行显示

FILE NAME“□

输入文件名(文件名可以任意取),例如“ABC”,按“Enter”键,则显示:

WREITING ABC

稍停片刻,又返回 SELECT 指令状态。

至此,图形存储完毕。切断录音机电源。

3. 从录音磁带中取出(读出)数据资料

操作步骤:

(1)取出(读出)BASIC 语言设计的程序,必须使系统处于 BASIC 状态。

• 从键盘输入 LOAD“文件名”指令。按“Entre”键。

• 按下录音机的放音键(PLAY),系统便能自动查询,取出(读出)该文件名的程序。(若省略文件名,则取出(读出)的是第一个找到的文件。

(2)取出(读出)BG GRAPHIC 的图形时,必须使系统处于 BG GRAPHIC 工作状态。

• 按“Esc”键,选择:“FILE” 指令控空格键,系统进入 FILE 指

令状态。。显示：

SAVE(S),LOAD(L)？□

- 键入“L”，在第二行显示

FILE NAME“□”

- 键入欲取出的文件名，按录音机的放音键(PLAY)、按“Enter”键。当系统读到文件时会显示

LOADING 文件名

过一段时间，即显示要取出的绘图。

- 若不键入文件名，则取出的是第一个找到的绘画。

至此，取出操作结束。切断录音机电源。

第二章　F BASIC 语言初步

语言是人类进行思想交流的工具。概括地说，它是由词汇和语法组成的符号系统。通常人们把这一符号系统称为自然语言，如汉语、英语、日语、法语等。这些语言已为人们所熟悉、所理解、所掌握、所应用。但是电子计算机却不能识别这种语言，它所能识别处理的只有计算机语言。

2.1　计算机语言

计算机语言有高级语言、低级语言之分，低级语言又分汇编语言和机器语言。机器语言是计算机的心脏 CPU（微处理器）所唯一能够识别和直接执行的语言。

2.1.1　机器语言

机器语言是由 CPU 能够执行的机器代码所构成的语言。使用机器语言的优点是不必经过翻译，CPU 可直接执行。它具有效率高，执行速度快等优点。但是，机器语言使用的机器代码是二进制数，我们知道二进制数仅由“0”、“1”两个数码表示，因而用机器语言编写的程序全部由“0”、“1”组成。显而易见，用机器代码书写的程序非常繁琐且极易出错。例如：我们将内存单元 $60 与 $61 中的数值相加（“$”表示后面的数字是 16 进制数），结果存入 $62 单元中，其 6502 机器语言程序为：

```
10100101
01100000
```

01100101
01100001
10000101
01100010

可见这种语言存在着严重的缺点：其一是二进制数看起来都差不多，尤其在大量阅读程序时，令人眼花缭乱、难以区别；其二是二进制码必须一个接一个的输入计算机，稍不留心就会出错，并且输入速度十分缓慢；其三是很难迅速理解电脑所要完成的是何种任务，执行的是何种操作；其四是程序冗长，编写麻烦，令人乏味，容易混淆，很难发现错误，发现错误纠正也极不方便；其五是记忆困难，人们很难一一记住所有指令的机器代码和数据单元地址以及所有的指令地址。

由于16进制数与二进制数有着简单的换算关系，所以为使程序书写简洁，人们采用了16进制数来表示二进制代码，这样上例的程序就可以写为：

A5 60 65 61 85 62

这种书写形式清晰多了。但是人们仍然难以区别何者是指令，何者是数据，何者是地址；也难以知道程序在做什么，到哪里去；难以知道A5、65各代表什么；而且不同类型的微处理器，其代码的含义还有所不同。所以，使用16进制数书写程序仍然不能解决机器语言程序阅读理解困难的问题。因此，人们继续探求比较接近自然语言的计算机语言来编制程序。从而出现了汇编语言。

2.1.2 汇编语言

汇编语言也叫符号语言，它是一种面向机器的程序设计语言。它的最大特点是用符号来代替机器码，这样书写起来比使用机器指令代码要方便的多。为便于记忆，操作码一般采用英文单词(或缩写)和符号来表示，这些单词和符号称为助记符。助记符可以反映指令的功能和主要特征。上例程序若用助记符表示则为：

```
LDA   $60
ADC   $61
STA   $62
```

这个程序功能虽然还不够明显，但至少其中某些部分已变得较容易理解了。如第一条指令A5的助记符是LDA，它是英文Load Accumutator（装载累加器）的缩写，累加器是CPU中的一个寄存器，记为A。指令LDA $60的意思是取$60存储单元中的数据送入寄存器A。在6502指令系统中，采用"$"号来表示后面的数字是16进制数编址的存储单元。第二条指令65的助记符是ADC，它是英文Add to Accumulator with carry（连同进位标志位与寄存器A中的内容相加，结果送入A中保存）的缩写。指令ADC $61的意思是取$61内存单元中的数据与进位标志C，与累加器A中的数据相加，相加之和仍存入A中。第三条指令85的助记符是STA，它是Store Accumulator to memory（把累加器中的内容存到存储器中）的缩写。指令STA $62的意思是把寄存器A中的数据送入存储器$62单元中。

可见，在汇编程序中，我们可以知道哪个部分是指令，哪个部分是数据，哪个部分是地址了。

但是，汇编语言终究是一种接近机器语言的低级语言，机器语言所具有的那些缺点（如繁琐、易出错、不够直观等）汇编语言同样存在，只不过程序上不同罢了。人们希望找到一种和自然语言更接近的并能为机器所接受的语言，这种语言我们称它为高级语言。

2.1.3 高级语言

一般人们把接近于机器语言的计算机语言称为低级语言，把接近自然语言的计算机语言称为高级语言。

计算机的高级语言是一种适合人的思维表达方式、接近英语自然语言并能被计算机所接受的语言系统，其表达方式接近于被描述的问题。高级语言是由不同的"关键字"和"表达式"按照一定

的语义和语法规则组成的。如欲求 35 与 25 两数之和，用 BASIC 语言编制的程序如下：

```
10 A=35
20 B=25
30 C=A+B
40 PRINT C
```

可以看出，高级语言完全脱离了具体计算机的指令代码系统，因此，用高级语言编写程序书写方便、简洁易读、容易修改。

通常把用高级语言编写的程序称为源程序，计算机的 CPU 不能执行源程序，它首先要经过该型机上配置的该种高级语言的编译程序或解释程序的翻译，生成目标程序(即机器码程序或机器语言程序)，然后才能执行这个程序。虽然经过翻译系统这一中间环节增加了运行时间，占用了内存空间，但却大大提高了高级语言程序的通用性。这一特点正是人们所孜孜追求的。

目前世界上已有几百种不同功能的高级语言，其中得到广泛应用的有十几种，最常用的有 BASIC、FORTRAN、PASCAL、COBOL、C 语言等。

2.1.4 BASIC 语言

BASIC 语言是目前国际上最通用的计算机语言。BASIC 是 Beginner's All-purpose Symbolic Instruction Code(初学者通用符号指令代码)一词的缩写。顾名思义，就知道它是一种适合初学者使用的计算机高级语言，它的特点是语法结构简单、容易掌握，是一种非常实用的会话式语言。所谓"会话式"是指当源程序输入计算机后，计算机马上检查输入是否有错，如果有错，它马上显示出错误信息，指出属于什么类型的错误，错在什么位置，用户可以立即在键盘上加以修改。另一方面，用户可以通过键盘随时干预程序的运行，对程序语句、中间结果进行检查修改。这种人机对话功能使得程序可以在计算机上边输入、边运行、边修改，这对初学者来说是

十分方便的。另外，它还可以不编程序利用键盘象电子计算器那样进行立即计算。如求 35 与 25 的和，可直接从键盘输入：

```
PRINT 35+25
```

它不仅可以作数学计算，还可以用于数据处理、事务管理和自动控制。通过学习 BASIC 语言作为高级语言的入门，在此基础上再学习另外的高级语言将十分容易。

2.2 F BASIC 语言浅述

2.2.1 F BASIC 语言的概念

BASIC 语言有许多版本，不同型号的机器其 BASIC 语言的结构也略有不同。家庭电视游戏学习机使用的 BASIC 语言也有许多版本，科特 FCS 90 电脑游戏机使用的 BASIC 语言是从日本 MSX BASIC 语言扩展、移植而来的，称为 FAMILY BASIC(家用 BASIC)语言简称 F BASIC 语言。

F BASIC 语言除符合基本 BASIC 语言的指令、语法、逻辑运算、程序设计等一般的通用规则及标准外，其显著特点是在简化指令的前提下，利用最简单的程序，最少的内存容量完成绘图和动画设计。

上例程序是进行两数求和运算。为加深认识，下面再列举一个完整的 F BASIC 程序示例。我们知道，在电子电路中计算流过三个并联电阻的总电流 I，其计算公式为：

$$I=U(1/R1+1/R2+1/R3)$$

现已知：U=12V

R1=10Ω

R2=20Ω

R3=40Ω

求 I 的程序可写为

```
10 R1=10
20 R2=20
30 R3=40
40 U=12
50 I=(1/R1+1/R2+1/R3) * U
60 PRINT I
70 END
```

这个程序的头四行是提供原始数据，第五行是计算 I，第六行是将计算结果显示打印出来，最后一行是说明程序到此结束。由此可见，程序是由一些英语词汇或缩写和一些数学表达式来表示的。我们学习计算机语言就是要学会这些表示方式，知道哪些符号是合法的、数据如何书写、函数怎样命名、有些什么句型及句型的含义和书写格式、一个程序应如何构成以及编程技巧等等。

2.2.2 F BASIC 语言所用的字符

F BASIC 语言中规定使用如下字符：

1. 26 个英文字母，有大、小之分，一般使用大写形式。
2. 0～9 十个阿拉伯数字。
3. 算术运算符

   “+”(加)、“－”(减)、“ * ”(乘)、“/”(除)、“MOD”(求余数)
4. 关系运算符

   “＝”(等于)、“＞”(大于)、“＜”(小于)，“＞＝”(大于等于)、“＜＝”(小于等于)、“＜＞”(不等于)。
5. 逻辑运算符

   “AND”(与)、“OR”(或)、“NOT”(非)、“XOR”(异或)。
6. 分隔符：

   “,”(逗号)、“:”(冒号)、“″”(双引号)、“ ’ ”(单引号)、“;”(分号)、“－”(连号)、“.”(点号)。
7. 其它符号

“$”(美元号)、“#”(半音)、“&”(16 进制符)、“?”(问号)。

2.2.3 保留字

F BASIC 中使用的语句、命令、函数和系统变量均称为保留字,用户不能使用保留字作为变量名或作为变量名中的一部分。保留字见表 2—1。

表 2—1 F BASIC 语言的保留字 (V3.10)

AUTO	AND	ABS	ASC
BACKUP	BEEP	BGGET	BGPUT
BGTOOL	CALL	CAN	CGSET
CHR $	CGEN	CLICK	COLOR
CLEAR	CLS	CRASH	CONT
CSRLIN	CUT	DATA	DEF
DELETE	DIM	END	ERA
ERL	ERR	ERROR	FILTER
FIND	FOR	FRE	GAME
GOSUB	GOTO	HEX $	IF
INKEY $	INPUT	INSTR	KEY
LEFT $	LEN	LINPUT	LIST
LOCATE	LOAD	MID $	MOD
MOVE	NEW	NEXT	NOT
OFF	ON	OR	PAUSE
PALET	PEEK	PLAY	POKE
POS	POSITION	PRINT	READ
RND	REM	RENUM	RESUME
RESTORE	RETURN	RIGHT $	RUN
SAVE	SCREEN	SCR $	SGN
SPC	SPRITE	STEP	STICK
STOP	STRIG	STR $	SWAP
SYSTEM	TAB	THEN	TO
TR	VIEW	VAL	VCT
XOR	XPOS	YPOS	

2.2.4 数据类型

F BASIC 使用的数据有两类:一类是数值型数据;一类是字符

串型数据(简称串型数)。

1. 数值型数据

数值型数据为整型数。整型数可以是正整数或负整数。F BASIC 能处理的数据最小的整数是－32768,最大的整数是＋32767。F BASIC 不处理实型数,即不能处理小数(仅在算术运算时可以处理小数)。若输入小数时,计算机会自动去尾取整,舍去小数点后的数字,且不遵循四舍五入的原则。

2. 串型数据

用双引号括起来的一串字符叫字符串,字符串可以作为数据参与运算和处理。字符串中的字符可以是字符集里除引号外的所有字符(由于引号要作为字符串的标志,所以字符串中不能再包含引号)。一个字符串最长不能超过 31 个字符。下面这些都是字符串数据:

"FCS－90 IS A MICROCOMPUTER"

"X,Y,Z="

"(50X－3A)/6A="

"GREAT CHINA!"

"＊/＋－$ #"

"35AZ#217"

一个空格也占据串型数中的一个位置。若一个串型数全部是空格则称为空串,空串也是串型数。

2.2.5 变量

F BASIC 有两类变量:简单变量、下标变量。简单变量又分整型数变量和字符串变量两种。一般把整型数变量称为变量,把字符串变量称为串型变量或串变量。

简单变量的名称可以用一个、二个、三个……字符表示,其表示规则为:

×××

最后一个字符表示变量类型。“$”表示串变量。没有“$”的为变量。

第二个字符可以是字母或数字，也可没有。

第一个字符必须是大写字母。

除了变量类型符号“$”之外，变量名也可以取多个字符，最多可取到255个(只能是字母和数字)。计算机会按照你命名的变量名称存储和打印赋予变量的值，但运算时它只识别除“$”号外的前两个字符。如：ABC1 和 ABC2 代表同一个变量“AB”，EOP 与 EOK 是同一个变量“EO”。然而 EOOP 与 EOOP$ 则不同，EOOP 是数值变量，EOOP$ 是串变量。

F BASIC 中的保留字不能做变量名，也不能在变量名中包含保留字。如：

```
ONAGOTO
```

计算机会把它作为 ON A GOTO 对待，这是因 ON、GOTO 均为保留字的缘故。

应用中要注意变量的类型要与数据类型保持一致。整型数变量只能取整型数，字符串变量只能取串型数。如：

```
AB=3215
A$="ABCEDFG"
```

以后还会看到，下标变量所能取的值也必须是数值或字符串。下标变量留待第三章讨论。

2.2.6 运算符号及运算规则和表达式

运算符号分算术运算符号、关系运算符号和逻辑运算符号三种。

1. 算术运算符号

算术运算符共有五种，它们的表达形式和规则见表 2—2。

表 2－2　算术运算符号表

运算符号	运算内容	示例	数学表达式
＋	加	A＋B	A＋B
－	减	A－B	A－B
*	乘	A * B	A×B
/	除	A/B	A÷B 或 A/B(B≠0)
MOD	求余数	A MOD B	A÷B 后的余数

• MOD 为计算“除”运算时的余数。

如：18÷4＝4……2

则有 18 MOD 4＝2

• 运算的优先顺序：

①“ * ”、“/”

②MOD

③“＋”、“－”

在同一算式中，若两个运算符号的优先级相同，则遵循先左后右的原则。若要改变运算的优先顺序，应将先运算的内容用括号“()”表示；如表 2－3。

表 2－3　数学表达式的表示方式

数学中表示	F BASIC 中表示
AX＋Y	A * X＋Y
$\frac{X+Y}{2}$	(X＋Y)/2
A[X＋B(X＋C)]	A * (X＋B * (X＋C))
$\frac{A}{B}\div\frac{C}{D}$	(A/B)/(C/D)

2.　关系运算符号

关系运算符号用于比较两个常量或变量时，计算结果如满足条件则其值为－1，否则其值为 0。

在 F BASIC 中，关系运算符号见表 2－4。

表 2—4　关系运算符号表

运算符号	运算内容	示例	数学表达式
==	两边相等	A=B	A=B
<>	两边不等	A<>B	A≠B
>	左边大于右边	A>B	A>B
<	左边小于右边	A<B	A<B
>=	左边大于或等于右边	A>=B	A≥B
<=	左边小于或等于右边	A<=B	A≤B

在 F BASIC 程序中，关系运算符号可用作条件判断语句。

例如：IF X>0 THEN 100

含义：如果 X 大于零，则跳到程序标号为 100 号语句执行。

3. 逻辑运算符号

逻辑运算符号是用于二进制中“0”及“1”的逻辑运算或其它二进制操作的记号，含义如表 2—5。

表 2—5　逻辑运算符号表

运算符号	运用内容	示例	逻辑表达式
NOT	否定(非)	NOT A	$\overline{A}>A$
AND	逻辑积(与)	A AND B	$A \wedge B$
OR	逻辑和(或)	A OR B	$A \vee B$
XOR	逻辑排它和(异或)	A XOR B	$(A \wedge \overline{B}) \vee (\overline{A} \wedge B)$

逻辑运算优先顺序：

①NOT

②AND

③OR

④XOR

逻辑运算真值表如下：

NOT

A	NOT
0	1
1	0

AND

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

XOR

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

注：变量与逻辑运算符号连续书写时，中间要留有间隔。

示例

①NOT 13

∵13=(0000 1101)$_2$

∴NOT 13 =(1111 0010)$_2$

=−14(补码)

②50 AND 5

∵50 =(0011 0010)$_2$

`5 =(00000101)$_2$

∴50 AND 5=(0000 0000)$_2$=0

注：(0000……)$_2$ 表示二进制位数。

4. 运算规则

运算规则与普通代数中的运算规则一样，如果表达式中有括号，则先算括号的内容，然后是函数、乘除、MOD、加减、关系运算、逻辑运算。它们的优先顺序是：

① 以括号()包括的部分

② 函数

③ *、/(乘、除)

① MOD (余数)

⑤ +、- (加、减)

⑥ =、<>、>、<、>=、<=

⑦ NOT (否定)

⑧ AND (逻辑积)

⑨ OR (逻辑和)

⑩ XOR (逻辑异或)

F BASIC 中的括号无大、中、小之分，一律用园括号()表示，可以在括号中再套括号。如

((((A+B*C)+((C+B)*C))*D)/C

2.2.7 记号及特殊用途

F BASIC 语言加强了记号(符号)的使用，从而大大简化了 F BASIC 语言的书写空间。各记号的含义如表 2-6。

表 2-6 记号与用途对照表

名称	记号	含义	示例
连号	-	在 LIST 等指令中，由于程序可能超过 24 行，可用"-"来指示电脑分段列出	LIST 100 - 300
逗号	,	在 PRINT、INPUT 等指令中，当超过一个操作数时，每个数之间需用"," 分开，以使电脑接受并继续操作。	PRINT A,B,C INPUT A,B,C DATA 10,13,9,8
冒号	:	同一程序中有 2 组以上指令时使用	A=B-C:PRINT A
分号	;	在 PRINT 等指令中，要作多个操作，两个数中用";"隔开，以使电脑了解执行先后。	PRINT "ABCD";A
问号	?	用"?"代替 PRINT 以节省空间	? A 与 PRINT A 相同
	&H	表示 16 进制数。	&H2C 即十进制的 44

2.3 F BASIC 的指令和程序

2.3.1 指令的分类

我们把 F BASIC 语言所有的关键字(即 F BASIC 语言的词)统称做指令。F BASIC 语言的指令共分五大类:命令、语句、函数、系统变量和算符。

1. 命令

命令为立即执行的指令,它是帮助用户对程序或计算机系统进行某种操作的一类词。如 LIST 可以对程序内容进行显示,RUN 可以运行程序等。命令不必写入语句行,计算机可以立即执行它所规定的操作。

2. 语句

语句是处在程序中、能够完成某种操作或实现某项具体功能的一类词。例如 INPUT、RETURN、LOCATE 等都是语句。F BASIC 语句往往也可以象命令一样以直接方式单独使用。它的作用对象一般不是 BASIC 程序。语句也称为暂缓型执行指令,一般写在程序中,待程序运行时才执行其规定的操作。

另外,本书还使用广义的语句或语句行的概念。在那里,语句是指程序中带有标号的一整行,而不是单个语句定义符。

3. 函数

F BASIC 通常把 ABS(X)、RND(X)等经常使用的数学函数预先编制成一个微程序供用户以后直接调用。用户调用这种函数时,只要给出微程序名并赋以自变量很快就可得到计算结果。这种微程序称为函数。F BASIC 语言不只包括数值函数,还包括用于处理文字的字符串函数及对数字与字符进行转换的转换函数。函数在程序中一般不能单独使用,通常只能作为语句行的一部分使用。

4. 系统变量

所谓系统变量，通常是指由 F BASIC 系统而不是由用户赋值的变量。如 ERR、ERL 等。

5. 算符

算符主要用来执行算术运算或者逻辑运算、进行关系比较或者对语句进行修饰说明。有些算符取英语中的某个词，即与普通单词形式相同，如 AND、NOT 等；有的则用非字母的符号表示，如“*”、“/”等。

一般算符不单列条目介绍（仅在使用中随文介绍），本书介绍的指令主要是指前四种。

2.3.2 指令的叙述方式

本文介绍的指令一般采用如下的叙述方式：

1. 格式

格式是指各个指令的使用格式。其中包括指令、参数、虚拟变量、必选项与可选项。有些指令同时具有两种以上的用途和用法时，一般分开列出。下面是格式中常用的几个符号。

“[]”方括号的内容为可选项，语句中可以不写。

“()”园括号要写在语句中。

“<>”尖括号表示语句体，语句中不要写。

“…”表示还可选用若干个同类型的可选项。双引号要写在语句中。

2. 类型

类型说明该指令属于命令、语句、函数、系统变量中的哪一种或两种。

3. 简写

简写表示计算机可以接受的指令的缩写。每个指令的缩写后面都要跟一个园点“.”；它是指令缩写的标识符。在使用指令缩写时一定要带园点。如 INPUT 可写为“I.”。

4. 功用

功用系指某一指令在其特定的格式下的功能及用途。该栏只对基本功能进行简要介绍，必要时还要介绍设立本条的背景和用意。具体功能在注释及说明中再随时解释。

5. 注释

对格式中所涉及的各项参数、变量、必选项及可选项的含义、用法、限定等进行具体说明。同时还对词条的来源和英文含义进行解释。

5. 示例

示例用于简要说明指令使用格式或功能。

6. 说明

主要用来对指令的功能与作用作进一步的补充指出注意事项。

2.3.3 语句和程序

先举一个用 F BASIC 语言解题的程序示例

例 2·1 设五年级地理课程的成绩，100 分的(A 等)10 人；90 分的(B 等)18 人；80 分的(C 等)21 人；70 分的(D 等)5 人；求全班的平均成绩。

用 F BASIC 语言编写程序如下：

```
10 REM "NO. 2—1"
20 LET A=10
30 LET B=18
40 LET C=21
50 LET D=5
60 LET G=100 * A+90 * B+80 * C+70 * D
70 LET U=A+B+C+D
80 LET F=G/U
90 PRINT F
100 END
```

这是一个完整的 F BASIC 程序，可以看出，它是由若干行组成

的，每行写一条语句，每条语句让计算机完成一定的操作，若干有序语句的集合就构成了一个程序。从上例可以清楚地看到一条语句一般可分为三部分：

1. 标号

每个语句前都要有一个数字(例 2.1 中的 10～100)，这个数字叫做行号或语句标号。行号是无符号整数。一般情况下，计算机按行号的大小从小到大顺序执行各语句。行号的取值范围为 0～65534，不要求连续(一般上、下行间都留出一定间隔以便修改、补插新的语句行，如程序 NO. 2－1 中取 10、20、30……；上下行间留出 9 行的间隔。当然也可以取 1、2、3、4……。)。各语句输入计算机时，不一定按行号大小顺序输入，行号杂乱的语句输入内存后，计算机会自动按行号大小顺序排列整理好。

2. 语句定义符

语句定义符就是计算机指令，它规定计算机执行一定的操作。如上例中的 PRINT、END。

3. 语句体

语句体是跟在语句定义符后面的、需要去执行的具体内容。如上例中的 A＝10 等。

每个语句不一定都有这三部分，但行号和语句定义符一定要有。每个程序一般以 END 结束，当后面没有别的语句时允许省略 END。

2.3.4 语句行规则

1. 一个语句行以标号开始，以回车键被按下为结束。

2. 一个语句行内最多允许有 255 个字符，其中包括键入的空格。

3. 一个语句行内可有若干个语句，每个语句之间用“:”分隔. 例如：

```
10 LET A=10:LET B=18:LET C=21:LET D=5
```

4. 通常要求语句定义符与变量之间留出一个空格。如

IF A=10 THEN 100

不要写为：

IFA=10THEN100

（没有空格的情况下，有时容易产生错误。）

2.4 F BASIC 的常用指令

作为 F BASIC 语言的入门，本章介绍六条最常用的指令。

2.4.1 变量赋值指令（“”）

格式： ＜变量＞=＜表达式＞

类型： 语句

功用： 在程序中为变量赋值。

没有赋值的数值变量其值通常为 0，没有赋值的串变量通常为空串“”。

注释：空格 LET 的省略写法。LET 在英文中为令或使的意思。

表达式是一个由变量和各种算符组成的数值型或字符型的算式，F BASIC 在执行此语句时，先对该式的值进行计算，然后将计算结果赋给符号左边的变量。如果表达式本身是个变量或常数，则其值将直接赋给左边的变量。另外，表达式还允许为逻辑运算式或逻辑运算结果，还可以是关系运算式。

当表达式为数值型时，其取值范围为－32768～＋32767。当表达式为字符串型时，其字符个数要小于或等于 31 个。

说明：

1. LET 语句具有计算功能。如程序 No. 2－1 中的 60 行、70 行分别计算了等号右边表达式的值，并分别赋给变量 G 和 U。

2. 语句说明符 LET 可以省略，即仅用等号就可对变量赋值。因此，程序 No. 2－1 可改写如下：

```
10 REM"No. 2－2"
20 A=10:B=18:C=21:D=5
30 G=100*A+90*B+80*C+70*D
40 U=A+B+C+D:F=G/U:PRINT F
50 END
```

3. 赋值语句中的等号与数学上等号有着本质的差别。在数学中

X+Y=Z

表示符号两边完全相等。也可以写为

Z=X+Y

两种形式的意义是相同的。而在赋值语句中的

Z=X+Y

则是指把 X、Y 的当前值相加后赋予变量 Z 而不管当时 Z 的值是多少，此语句执行后 Z 得到一个新值。如：

X=13 Y=14 Z=15

若写成：

Z=X+Y

在数学中显然是不成立的，但在赋值语句中则是正常现象。该语句执行后，由 Z=15 变为 Z=27。再如

X=X+1

是经常使用的赋值语句，它表示把 X 的当前值加 1 之后再送给 X，作为 X 的新值。这在数学中是不行的。因此，读者要特别注意这一差别。

4. 赋值语句可以对串变量赋值。如：

A$="FCS－90 COMPUTER"

此句是将右边的字符串送入左边的串变量中，此时 A$ 获得了新值。A$ 的新值就是一个确定的字符串"FCS－90 COMPUTER"。

注意：字符串都要带上双引号。且字符串只能赋给串变量，数值只能赋给数值变量，两者不能交叉赋值。

5. 赋值语句的左边只能是一个变量，不能是多个变量或表达式。如只能写作：

Z=X+Y

而决不能写作：

X+Y=Z

同样

4*Y=20

也是错误的。

2.4.2 运行程序指令(RUN)

格式： RUN[行号]

类型： 命令

简写： R.

功用： 运行内存中的当前程序。

注释： RUN 在英文中为运行的意思。

如果在 RUN 中选用行号，则程序只从该行号所指定的语句开始向下运行。若不选用行号，则从程序的第一行开始运行。

示例：RUN 50

说明：RUN 命令执行时，首先清除内存中现有程序中的变量。执行一次 RUN 后，程序并没有被破坏，可以多次执行。

2.4.3 打印输出指令(PRINT)

格式： PRINT＜数据＞

类型： 语句或命令

简写： P. 或?

功用： 对数值变量或串变量的内容进行显示。

注释： PRINT 在英文中为打印的意思。

1. 数据是指被显示的内容，可以是数值型的，也可以是字符串型的。无论数型还是串型，既可取常数形式，也可取变量形式，还可以取表达式形式。如果数据取表达式形式，则 PRINT 先对表达

式进行计算，然后再显示计算结果。

2. PRINT 语句可以输出多项数据。当数据超过两项以上时，需要用逗号(,)或分号(;)分隔开。逗号和分号的作用是不同的。逗号可以使数据各项按分区方式显示，即每一显示行按固定间隔划分成若干个区域，数据项只能显示在各区域内；分号可以使数据以紧凑格式显示。

示例 1：

```
10 REM"PRINT1"
20 A=100:B=200
30 A$="HOW ARE YOU。"
40 PRINT A$,A,(A+B)/2
50 END
RUN  ↙
```

("RUN ↙"表示键入 RUN 指令，并按回车键。"↙"表示按回车键，以后不再说明。回车键即"Enter"键，也称输入键。)

程序运行后，立即显示：

```
HOW ARE YOU.    100    150
OK      (OK 为程序执行完毕标志)
```

由于输出的各数据之间采用了逗号，所以显示的内容是分区的。

说明：

PRINT 指令可以控制多种显示格式，最常用的是分区显示、紧凑显示、单列显示和隔行显示四种。

1. 分区显示

分区显示也叫标准显示格式。F BASIC 的屏幕显示行，一行可以显示 28 个字符。在采用分区显示时，计算机会根据显示项字符的多少，自动分成四个区、或三个区或两个区。如果最后一个区的空间显示不下，剩余的部分会自动移到下一行的第一区显示。逗号","是分区显示的标识符。

2. 紧凑显示

紧凑显示的标识符是分号“;”。

示例 2

```
10 REM "PRINT2"
20 A$ ="GOOD":B$ ="MORING":C$ =" * "
30 PRINT A$ ;C$ ;B$
40 END
RUN          ↙          显示
GOOD * MORNING
```

可见，输出项之间使用分号后，变为紧凑显示。

在紧凑显示时，F BASIC 允许省略标识符“;”。如示例 2 中的 30 句可写为：

```
30 PRINT A$ C$ B$
RUN          ↙          仍显示
GOOD * MORNING
```

对于数值型数据，当以紧凑格式显示省略标识符时，每输出项间要留一空格。

示例 3

```
10 REM "PRINT3"
20 A=10: B=20: C=30
30 PRINT A;B;C
40 END
RUN          ↙          显示
10 20 30
```

若把 30 行改为

30 PRINT A B C （“ ”表示空格）

```
RNU          ↙          同样显示
10 20 30
```

如果 30 行 A、B、C 中间不留空格，如 30 PRINT ABC

```
RUN        ↙        则会显示
0
OK
```

这是 F BADIC 把 ABC 看作是一个变量，由于变量 ABC 未赋值，所以显示 0。

对比示例 2、3 的运行结果，我们发现了一个问题，即同样是紧凑格式显示，当显示串型数时，显示项之间没有空格；当显示数值时，显示项之间却出现空格。这一现象是 F BASIC 为数值型数据予留符号位的设置造成的。当显示的数据小于 0 时则在其前面显示“－”号；若大于 0 时则显示一个空格。

示例 4

```
10 REM “PRINT4”
20 A=10 :B=－20:=－30
30 PRINT A B C
40 END
RUN        ↙        显示
10－20－30
```

现在显示已没有空格了。

3. 单列显示

若每个 PRINT 语句只显示一项数据，当多个语句顺序执行时，将采取单列显示方式。

示例 5

```
10 REM“PRINT5”
20 A=10 :B=20 :C=30
30 PRINTA:PRINT B:PRINT C
40 END
RUN        ↙        则显示
10
20
```

```
30
OK
```

4. 隔行显示

PRINT 指令后边可以不带参数，即没有数据项。PRINT 不带参数时，表示进行空显示，表现在屏幕上就是空一行。

示例 6

```
10 REM"PRINT6"
20 A=10:B=20
30 PRINT A
40 PRINT
50 PRINT B
60 END
RUN          ↙          显示
10

20
OK
```

5. 提示显示

在以上的示例中，PRINT 语句显示的是变量的值。当显示变量较多时，容易混淆变量与变量值的对应，因此人们希望能够对显示的内容进行提示。PRINT 语句可以"拷贝"显示字符串。所谓拷贝显示就是把 PRINT 语句中的字符串原样显示在屏幕上。

如 PRINT"A=20" ↙则显示

```
A=20
```

因此使用字符串可以达到提示显示的目的。

示例 7

```
10 REM"PRINT7"
20 A=10:B=20:C=30
30 D=A*B:E=A+B+C:F=C/A
40 PRINT"A="A,"B="B,"C="C,
```

```
"A * B="D,"A+B+C="E,"C/A="F
50 END
RUN          ↙          显示
A= 10        B= 20        C= 30        A * B= 200
A+B+C=60          C/A= 3
OK
```

6. 关于 PRINT 语句后标点符号的讨论

在示例 5 中我们已经看到当 PRINT 语句后没有标点时，每一个 PRINT 指令显示一行，若有标点时其显示形式又有变化。

示例 8

```
10 REM"PRINT8"
20 A=10:B=20:C=30
30 PRINT A; :PRINT B; :PRINT C
40 END
RUN          ↙          显示
10  20  30
OK
```

可见，当最后是分号时，各 PRINT 语句采用紧凑格式显示，此时相当于一条语句：

```
PRINT A;B;C
```

同理，若最后是逗号时，各 PRINT 语句采用分区格式显示。请读者验证。

7. PRINT 指令还可以以立即执行方式使用，即不用编入程序中，直接用键盘输入执行。如显示"I AM WORKING."。可键入

```
PRINT"I AM WORKING."。          ↙
```

立即显示上述内容。

同 LET 指令一样，PEINT 语句还具有计算功能。如

```
PRINT (30+20) * 50          ↙
```

立即显示 2500。

通过以上介绍可知，PRINT 指令的使用形式是多种多样的。实际上，在一切 BASIC 版本的指令中，PEINT 是最活跃的一条指令。它应用最广泛，使用最灵活。请大家进行反复练习，以求熟练掌握。

另外，PRINT 指令也称打印指令或输出指令，同样 PRINT 语句中的数据也称为输出项或打印项。显示、打印、输出对 PRINT 指令而言是同一个意思。请读者适应这一称谓。

2.4.4 程序列表指令(LIST)

当一个程序输入内存后或者运行后，如欲检查程序或对程序进行修改，可使用 LIST 指令列出源程序清单。

格式：LIST[[M][－n]]

类型：命令

简写：L.

功用：显示部分或全部程序语句内容，如果程序语句不是按行号顺序编辑的，LIST 命令可将程序按行号顺序整理后再显示出来。

当我们编辑一个程序或要了解程序内容时，就需要使用这条命令。

注释：LIST 在英文中为清单，列表的意思。

m 为欲列表的首行号，n 为欲列表的末行号。m、n 的取值范围为 0～65534。

m、n 为任选项，它们的选用和对应的功能如下：

LIST M——列出行号为 m 的一行程序

LIST M－——列出 m 行及 m 行以后的所有程序。

LIST －n——列出起始行至 n 行的所有程序。

LIST m－n——列出 m 行至 n 行的程序。

LIST——列出全部程序。

示例：

LIST 40—100 ↙

该语句执行后，程序 40～100 行语句的程序内容(包括第 40、100 行)就会显示出来。

LIST —100 ↙

该语句可以显示第 100 语句及之前的程序内容。

LIST 60— ↙

该语句可以显示第 60 行及以后的所有程序。

2.4.5 内存清除命令(NEW)

一个程序执行完毕，当欲进行新的程序设计或输入另外的程序时，必须将内存中的原程序清除掉，以免干扰新输入的程序。清除原程序的过程叫做清内存。清内存有多种方法，如关机后重新开机。常用的清内存方法是使用内存清除命令 NEW。

格式：NEW

类型：命令

功用：清除内存中的当前程序，将全部数值变量置零，将串变量置空。这一命令通常在输入新程序前，为了避免现有的程序对新程序干扰的情况下使用。

注释：NEW 在英文中为新的意思。在此表示把系统置为新的状态(即未使用过的状态)也即是清除内存。

示例：

```
10 REM"NEW"
20 A=10:B=15:A$="HELLO"
30 PRINT A,B,A$
40 END
RUN ↙ 显示
10 20 HELLO
```

运行结果表明程序有效。这时再使用 LIST ↙

则显示程序清单，证明程序在内存中。现在使用 NEW 命令

NEW ↙ 再键入 RUN 运行程序则有

RUN ↙ 仅显示

OK 再使用列表指令

LIST ↙ 也仅显示

OK （表示没有程序可显示。）

这时再键入

PRINT A,B,A$ ↙ 则显示

0 0

表明变量、程序、内存全部被清除。

2.4.6 程序注释指令(REM)

通常一个程序最少要解决一个问题，即使是最短小的程序，也要具有一定的功用，或完成一定的操作。但当程序个数较多时，往往记不清每一个程序的作用，若靠阅读程序找出功用则事倍功半，效率太低。因此人们希望对每一个程序起一个名字或对其功用做一简要说明，以便迅速确定功能，方便使用。另外在一些较复杂的程序中，为增强程序的易读性，也要求对某些程序段的功能进行简要说明。REM 指令就是为满足这一要求而设置的专用指令。

格式： REM<注释内容>

类型： 语句

简写： ’

功用： 对语句或程序段的作用作简要说明，以增强程序的易读性。

注释：REM 源于 Remark，是注释、注解或说明的意思。

注释内容可以是任何文字、字符，甚至可以是 F BASIC 的关键字(保留字)。注释内容仅仅是给编程序的人或阅读程序的人看的，程序运行后，F BASIC 对注释语句的内容不予理会，故 REM 为非执行语句，即计算机并不执行此语句。

注释内容以 255 个字符为限。

示例：

REM“PRINT”

REM“NO. 2—1”

以上所举例中的 10 行均为注释语句。

说明：

1. 在多重语句行(一个行号下含多条语句)中，只有 REM 之前的语句才能执行，而同行中 REM 后的语句全作为注释内容对待，不予执行。

2. 如对多个语句进行注释，则每个被注释的语句标号都要写上 REM。

3. 注释语句虽为非执行语句，但在用 LIST 命令显示程序清单时仍会显示该语句的全部内容。

4. 以后所学到的 GOTO、GOSUB 等语句可以转移到 REM 语句上，但从 REM 后第一个可执行的语句处开始向下执行。

本章重点介绍了 F BASIC 语言的起源、特点、有关约定及指令的介绍方式，并介绍了六条最常用的指令。从下章开始将以较多的示例系统地介绍 F BASIC 语句的其它指令。

第三章　F BASIC 语言的基本指令

第二章介绍了几个初步的 F BASIC 指令，从本章开始将陆续介绍 F BASIC 语言中使用最经常、最基本的指令。

3.1　循环语句

3.1.1　循环语句的结构

循环语句由 FOR、NEXT 两句组成。FOR 语句是循环的起点，是循环的说明语句，由它决定循环的次数。NEXT 语句是循环的终端语句。FOR 与 NEXT 语句必须成对出现。

FOR 语句的格式如下：

格式：FOR i=l TO m STEP S

类型：语句

功用：FOR—TO 为循环的起始语句，它可以确定循环变量的初值、终值、步长值，并给出循环变量增加的步长值。

说明：

i—循环控制变量，可用任一简单变量表示。

l—循环控制变量初值。

m—循环控制变量终值。

s—循环变量每次循环的增量，也叫做步长。步长为 1 时，可以省略 STEP S。

注释：FOR TO 语句的意思是：循环从变量等于初值开始，到

变量超出终值时为止。每循环一次，变量就增加一个步长值。循环的初、终值、步长一般为数值，也可以取变量形式。

例 3.1　打印 1～30 整数的平方、立方表。

```
5 REM "No. 3—1"
10 FOR A=1 TO 30 STEP 1
20 PRINT A,A * A,A * A * A
30 NEXT
RUN ↙     ("↙"表示按回车键)
```

则立即显示

```
1           1           1
2           4           8
3           9           27
.           .           .
.           .           .
.           .           .
.           .           .
.           .           .
.           .           .
29          841         24389
30          900         27000
```

因为步长为 1，例 3.1 程序中 10 句也可写为 FOR A＝0 TO 30。步长也可以是负值。当步长为负值时，终值应小于初值。

例 3.2　降序打印 20～2 之间的偶数。

```
5 REM "No. 3—2"
10 FOR X=20 TO 1 STEP —2
20 PRINT X,
30 NEXT
RUN ↙
```

```
20      18      16      14
12      10      8       6
```

4　　2

例 3.3　求阶乘 5!

```
5 REM "No. 3－3"
10 A＝1
20 FOR B＝1 TO 5
30 A＝A * B
40 NEXT
50 PRINT A
RUN ↙　打印
120
```

注意——循环变量的初、终值、步长必须控制在－32768～＋32767 之间取值。否则程序不能输入内存，屏幕显示"? OV ERROR"，表示溢出。

F BASIC 允许输入语句的简写形式。简写输入后，计算机会自动翻译成原语句。如 FOR—NEXT 语句可简写为 F. ～TO～ST. —N. 。注意简写中字母后的点不能丢失。读者要习惯语句的简写形式，以便提高程序的编写、输入速度。同样 LIST 的简写为 L. ；PRINT 为 P. 或?；END 为 E. 。但是为了便于理解，本书中的叙述仍使用原语句，不使用简写。

例 3.4　求自然数列前 100 项之和 $\sum_{n=1}^{100} n$

```
5 REM "No. 3－4"
10 C＝0
20 FOR D＝1 TO 100
30 C＝C＋D
40 NEXT
50 PRINT C
RUN ↙　显示
5050
```

3.1.2 多重循环

以上介绍的是单重循环，F BASIC 允许两重以上的多重循环嵌套。

例 3.5　打印九九乘法表

用二重循环编制程序如下：

```
5 REM "No. 3—5"
10 FOR A=1 TO 9
20 PRINT
30 FOR B=1 TO 9
40 C=A * B
50 PRINT A " * " B "=" C,
60 NEXT
70 NEXT
```

执行结果如下：

```
RUN ↙
1 * 1=1          1 * 2=2
1 * 3=3          1 * 4=4
1 * 5=5          1 * 6=6
1 * 7=7          1 * 8=8
1 * 9=9
2 * 1=2          2 * 2=4
2 * 3=6          2 * 4=8
2 * 5=10         2 * 6=12
2 * 7=14         2 * 8=16
2 * 9=18
3 * 1=3          3 * 2=6
3 * 3=9          3 * 4=12
3 * 5=15         3 * 6=18
3 * 7=21         3 * 8=24
3 * 9=27
```

```
⋮
⋮
9 * 1=9          9 * 2=18
9 * 3=27         9 * 4=36
9 * 5=45         9 * 6=54
9 * 7=63         9 * 8=72
9 * 9=81
```

程序中 20 行的作用是使运行结果分段打印。读者可去掉 20 行试试。

F BASIC 允许在一个语句行里写多个语句，语句之间用冒号隔开，只要一个语句行的字符总数不超过 255 个即视为合法。另外例 3.5 的程序运行结果重复部分较多，我们可以把循环变量 B 的终值改写一下，以删去重复部分。例 3.5 的合理程序如下：

```
5 REM "No. 3—6"
10 FOR A=1 TO 9 : PRINT : FOR B=1 TO A
20 PRINT B "*" A "=" B * A,
30 NEXT : NEXT
RUN ↙
1 * 1=1
1 * 2=2          2 * 2=4
1 * 3=3          2 * 3=6
3 * 3=9
1 * 4=4          2 * 4=8
3 * 4=12         4 * 4=16
1 * 5=5          2 * 5=10
3 * 5=15         4 * 5=20
5 * 5=25
1 * 6=6          2 * 6=12
3 * 6=18         4 * 6=24
5 * 6=30         6 * 6=36
```

1 * 7=7　　2 * 7=14
3 * 7=21　　4 * 7=28
5 * 7=35　　6 * 7=42
7 * 7=49
1 * 8=8　　2 * 8=16
3 * 8=24　　4 * 8=36
5 * 8=40　　6 * 8=48
7 * 8=56　　8 * 8=64

1 * 9=9　　2 * 9=18
3 * 9=27　　4 * 9=36
5 * 9=45　　6 * 9=54
7 * 9=63　　8 * 9=72
9 * 9=81

显而易见，改进后的运行结果确实令人满意。

使用多重循环嵌套时，要注意内外循环不能交叉，只能一层套一层，内层应完全介于外层的 FOR 语句和 NEXT 语句之间。如下图所示：

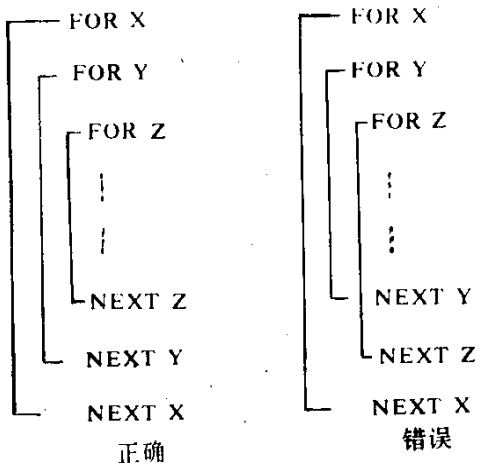


图 3—1

3.2 转移语句

计算机在执行程序时，一般是按行号大小由小到大顺序进行。但由于解决问题的需要，往往要求改变程序的执行顺序，转移到其它行号，这一现象叫做程序转移或者叫程序转向。实现这一功能的语句叫转移语句。转移语句有三种形式：

- 无条件转移语句(GOTO)
- 条件转移语句(IF—THEN)
- 开关转移语句(ON—GOTO)

3.2.1 无条件转移语句

格式：GOTO <行号>

类型：语句

简写：G.

功用：改变程序顺序执行的规则，将控制权交给规定行号的语句。

注释：GOTO 在 BASIC 中是“转向或转去执行”的意思。

行号为转移的目的地，代表所要转移的程序语句。行号要求取常数。

运行中当程序执行到这一语句时，把程序的运行无条件的转移到 GOTO 语句规定的行号去，并从这一行号起继续执行。

3.2.2 条件转移语句

格式：IF<关系式>THEN<行号>

IF<关系式>GOTO<行号>

IF<关系式>THEN GOTO<行号>

IF<关系式>THEN<语句>

类型：语句

简写：IF<关系式>T. <行号>

IF<关系式>G.<行号>

IF<关系式>T. G.<行号>

IF<关系式>T.<语句>

功用:根据某一关系式成立与否决定程序是否转移和转移的目的地。

IF—THEN 语句也叫 IF 语句,它可以改变程序顺序执行的规则。

注释:IF…THEN…为"如果……条件成立则……"的意思。

关系式是关系表达式或逻辑表达式。关系表达式是由<、>、=、<>等关系运算符连接起来的式子,在形式上类似于数学中的不等式或方程。逻辑表达式是由 AND、OR、NOT、XOR 逻辑运算符连接起来的式子。关系式也可以是关系表达式和逻辑表达式的混合形式。

如果关系式成立(即经运算后结果为真)则表明 IF—THEN 语句的判断条件成立,此时,BASIC 将会转到 THEN、GOTO 后的行号继续执行或执行 THEN 后面的语句。若关系式不成立,BASIC 就跳过本语句,执行后面的语句,此时相当于 IF 语句不存在。

IF 语句中,关系式后面可以用 THEN、可以用 GOTO、也可以用 THEN GOTO,三种形式效果一样,因此前三种格式可以任选。

例 3.6　用转移语句求自然数列前 100 项之和 $\sum_{n=1}^{100} n$

程序如下:

```
5 REM "No. 3—7"
10 A=0 : B=0
20 A=A+1 : B=B+A
30 IF A<100 THEN 20
40 PRINT "B="B : END
RUN ↙ 显示
5050
```

程序中 10 行对 A、B 变量清零，20 行进行连加运算，30 行为条件判断语句。当执行到 30 句时，计算机进行判断 A 是否小于 100，若 A 小于 100 则转到 20 行继续进行加法运算。若 A 等于 100 则关系式 A＜100 不成立，跳过 30 行，执行 40 行。在 30 行的关系式若换用 GOTO 20 或 THEN GOTO 20 效果是一样的。读者可自己验证。

例 3.6 的程序还可改为如下形式：

```
5 REM "No. 3—8"
10 A=B
20 A=A+1 : B=B+A
30 IF A=100 THEN PRINT "B=" B : END
40 GOTO 20
```

30 行中的条件关系式改为 A＝100，它表示若 A 等 100 时则执行 THEN 后面的语句，输出运算结果、停机。若 A 不等于 100 则跳过 30 行，执行 40 行。

通过例 3.6 可以发现，程序的编制是很灵活的，解决同一个问题，可以编制不同形式的程序。若欲求其它数值的前 n 项和，只须改变 30 行关系式中的数值即可。

由于循环语句、转移语句是程序设计中最常用、最有用的语句，同时也是最难全面掌握、最容易出错的语句。因此，本文将通过较多的示例介绍这两条语句的配合使用。并留一民间趣题由读者自己编程。

例 3.7　有一算式，式中每一种动物代表一个阿拉伯数字，试求出各代表的数字。

算式如下：

马蛇×马马＋牛牛＝1991

设 A＝马、B＝蛇、C＝牛，程序如下：

```
5 REM "No. 3—9
```

```
10 FOR A=0 TO 9 : FOR B=0 TO 9 : FOR C=0 TO 9
20 IF (A * 10+B) * (A * 11)+C * 11=1991 THEN PRINT "Ma="A,
   "She="B,"Niu="C : END
30 NEXT : NEXT : NEXT
RUN ↙      显示
Ma=4      She=3      Niu=9
```

THEN 后可以跟多个语句。BASIC 执行完第一个语句后再执行第二个语句……，直到执行完所有语句再执行下一行。20 行中 THEN 后的第二语句是 END，所以程序执行到此处就停机。

上述程序只求出了例 3.7 的一组解。可能还有其它解，这只要将 20 行的 END 语句移到程序的最后一行(添 40 行)，则可求出全部的解。

运行结果如下：

```
Ma=4      She=3      Niu=9
Ma=4      She=4      Niu=5
Ma=4      She=4      Niu=1
```

例 3.8　民间趣题

用 100 匹马驮 100 块瓦，大马驮三块，中马驮两块，两匹小马驮一块。求大马、中马、小马各用多少匹。

这是一个多解问题，先列出求解方程：

设大马为 X 匹、中马为 Y 匹、小马为 Z 匹，可列出方程组：

$$\begin{cases} X+Y+Z=100 \\ 3X+2Y+\frac{1}{2}Z=100 \end{cases}$$

整理得：

$$\begin{cases} X+Y+Z=100 \\ 6X+4Y+Z=200 \end{cases}$$

这道题若用代数方法求解，需用试商法，比较麻烦。若用计算机求解则比较简单。程序设计思路是：先设定单用一种马驮最多用

多少匹，从而决定循环变量的终值。可以算出大马最多用 34 匹；中马用 50 匹；小马用 100 匹(因出题限定 100 匹)。程序如下：

```
5 REM "No. 3—10"
10 FOR X=1 TO 34 : FOR Y=1 TO 50 : FOR Z=100
20 IF X+Y+Z=100 THEN 40
30 GOTO 70
40 IF 6 * X+4 * Y+Z=200 THEN 60
50 GOTO 70
60 PRINT "X="X,"Y="Y,"Z"=Z
70 NEXT : NEXT : NEXT
80 END
RUN ↙    打印
X=2      Y=30      Z=68
X=5      Y=25      Z=70
X=8      Y=20      Z=72
X=11     Y=15      Z=74
X=14     Y=10      Z=76
X=17     Y=5       Z=78
```

由于程序中有三重循环嵌套，执行速度较慢。若把 Z 的循环变量初、终值改为

```
FOR Z=60 TO 80
```

速度要快一倍。(由于小马 60 匹才驮 30 块，下余 40 匹马、70 块瓦，若全部用中马驮，也仅用 35 匹，所以小马最小不能低于 60 匹。同理，用 80 匹小马驮 40 块瓦，下余 20 匹马 60 块瓦，全部使用大马刚好驮完，所以小马最多不会超过 80 匹。)

例 3.9　山上一群羊，山下一群羊，若山上拿一只羊给山下，山上、山下羊数相等。若山下拿一只羊给山上，则山上的羊数是山下的两倍。求山上、山下各有多少只羊。

由于山上、山下羊数有一只的波动，就能产生倍级的量变，所以可断定山下、山上的羊数各不超过 10 只。

设山上有 A 只羊，山下有 B 只羊，可建立方程：

$$\begin{cases} A-1=B+1 \\ A+1=(B-1)*2 \end{cases}$$

程序如下：

```
5 REM "No. 3-11"
10 FOR A=1 TO 10 : FOR B=1 TO 10
20 IF A-1=B+1 AND A+1=(B-1)*2 THEN PRINT"Shan Shang="
   A,"shan Xia="B
30 NEXT : NEXT
40 END
RUN
Shan shang=7          Shan Xia=5
```

20 行使用了逻辑判断式 AND，它表示两关系式相与。即若两关系式都成立，则执行 THEN 后的语句；若两关系式中有一个不成立，则跳过 20 行而执行 30 行。使用逻辑判断式可使程序简洁。同样程序 No. 3—10 可以改编如下：

```
5 FOR "No. 3—12:
10 FOR X=1 TO 20 : FOR Y=1 TO 30 : FOR Z=60 TO 80
20 IF X+Y+Z=100 AND 6*X+4*Y+Z=200 THEN PRINT "X="X,
   "Y="Y,"Z="Z
30 NEXT : NEXT : NEXT
40 END
```

例 3.10　名人做过的趣题。美国物理学家爱因斯坦曾做过一道数学趣题，内容是：有一阶梯，每步跨两阶，剩一阶；每步跨三阶，剩两阶；每步跨五阶，剩四阶；每步跨六阶，剩五阶；每步跨 7 阶不剩。问阶梯至少应有多少阶？

程序如下：

```
5 REM "No. 3—13"
10 FOR N=0 TO 500 STEP 7
20 IF N MOD 2=1 AND N MOD 3=2 AND N MOD 5=4 AND N MOD 6
```

```
    =5 THEN PRINT N: END
30 NEXT
RUN    ↙      显示
119
```

No. 3—13 中使用于 MOD(求余数)运算。关系式中只判断余数如何而不管商是多少,只要各余数满足要求即得出问题的答案。

例 3.11　民间趣题

兄弟仨合伙打鱼,老大把打的鱼按条数三等分,最后剩了一条,由于一条不好分,老大把这条鱼扔进了河里,拿一份先回家了。老二不知道老大已分了鱼,就把剩下的两份鱼合在一起三等分,结果又剩了一条,他也把这条鱼扔进河里,拿走了一份。老三也不知道已分鱼,他又把剩下的两份合在一起三等分,也剩了一条,也扔进了河里。问至少有多少条鱼才能满足上述分法?

该题比较复杂,得数必须满足除以 3 余 1,其商乘 2 之积除以 3 余 1……。该题虽有无数个解,但用代数求解也大费脑筋。下面编程解决。

设得数为 A,循环终值为 100。

```
5 REM "No. 3—14"
10 FOR A =1 TO 100
20 B=A/3 : C=(A—B—1)/3
30 IF A MOD 3=1 AND 2 * B MOD 3=1 AND 2 * C MOD 3=1 THEN
   PRINT "A="A:END
40 NEXT
RUN ↙      打印
A=25
```

若求 100 以内的解,只须把 30 行末的 END 语句删掉,运行程序可求出三个解,分别为:25、52、79。欲求 1000 或 10000 之内的解则只须把循环终值分别改为 1000 或 10000 即可。

在 IF 语句中,THEN 后面的语句也可以是 IF 语句,这称为 IF

语句嵌套。因此程序 No. 3－14 中的 30 行也可以写成如下形式：

```
30 IF A MOD 3=1 THEN IF 2*B MOD 3=1
   THEN IF 2*C MOD 3=1 THEN PRINT "A=";A,
```

两种形式，结果相同。请读者验证。

注意——在使用 IF 语句嵌套时，IF—THEN 必须成对出现。

例 3.12　有一些三位数字，每位数字的取值可为 0、1、2、3，但最高位不能为 0。试求出满足这一条件的所有三位数。

解决这一问题可用三重循环实现：

```
5 REM ;No. 3－15;
10 FOR A=1 TO 3 : FOR B=0 TO 3: FOR C=0 TO 3
20 PRINT A;B;C,
30 NEXT : NEXT : NEXT
40 END
RUN ↙     打印
100     101     102     103
110     111     112     113
120     121     122     123
130     131     132     133
200     201     202     203
210     211     212     213
220     221     222     223
230     231     232     233
300     301     302     303
310     311     312     313
320     321     322     323
330     331     332     333
```

若要求一个数字的三位之间不能重复，即不允许出现 331、100、113 等之类的数字。可把程序 No. 3—15 改写如下：

```
5 REM "No. 3—16"
10 FOR A=1 TO 3 : FOR B=0 TO 3
```

```
20 IF A=B THEN 70
30 FOR C=0 TO 3
40 IF A=C OR B=C THEN 60
50 PRINT A;B;C,
60 NEXT
70 NEXT
80 NEXT
RUN      ↙      打印
102      103      120      123
130      132      201      203
210      213      230      231
301      302      310      312
320      321
```

例 3.13　民间趣题

龟、蟾、蟹共 19 只，100 条腿进行分配。求各有多少只？(注：龟四条腿，蟾三条腿，蟹十条腿)该题留给读者练习。

3.2.3　开关转移语句

格式：ON＜表达式＞GOTO＜行号 1＞，＜行号 2＞，＜行号 3＞……

类型：语句

简写：O. ＜表达式＞G. ＜行号 1＞……

功用：开关转移语句起分支转移作用，它根据表达式的值的大小从多个顺序排列的转移分支中选取其中一个，从而控制程序的转向。ON—GOTO 语句实际上是若干个连续的 IF—THEN 语句的合并。

注释：ON—GOTO 的意思是“当……时则相应执行……语句”。

表达式的值一般为数值，其取值范围为 1～255。

语句中的行号代表各转移分支。各个行号的排列位置与表达

式的取值大小是顺序对应的。当表达式的值为 1 时，BASIC 转去执行行号 1 所示的程序语句；当表达式的值等于 2 时，则转去执行行号 2 所示的程序语句……。余者以此类推。

当表达式的值等于 0 时或因过大而无行号与之对应时则不做转子处理，而是顺序执行 ON—GOTO 的下一语句。

当表达式的值大于 255 或小于 0 时则打印。

? IL ERROR IN ××× （×××表示行号）表示叙述错误。

开关转移语句还有另外三种形式：

ON＜表达式＞GOSUB＜行号 1＞,＜行号 2＞,＜行号 3＞,……

ON＜表达式＞RETURN＜行号 1＞,＜行号 2＞,＜行号 3＞,……

ON＜表达式＞RESTORE＜行号 1＞,＜行号 2＞,＜行号 3＞,……

开关转移语句的用处很大，我们将在介绍 INPUT 语句时详细举例。

3.3 键盘输入语句

3.3.1 INPUT

格式：INPUT“提示符”;变量 1,变量 2,变量 3,……

类型：语句或命令

简写：I.

功用：接收来自于键盘输入的数据并把它们赋给相应的变量。

INPUT 语句虽然与 LET 一样，同样用于给变量赋值，但它不是在编程时给变量赋值的，而是在程序运行时由用户根据实际需要从键盘给变量提供数据的。因此它所接收的数据对程序具有独

立性。这一语句适用于在变量取值事先无法确定或同一程序处理不同数据的场合。

注释:INPUT 在英文中为输入的意思。

语句中提示符是由用户设定的提示字符串,这些字符串必须使用双引号。当程序执行到 INPUT 语句时,屏幕上就会显示出这些字符信息,提示用户输入相应的数据。若在输入语句时省略了提示符和后面的分号,则执行到该语句时只显示一个问号,无其它提示信息。

变量就是 INPUT 语句所要赋值的对象,数值或字符串均可。一个 INPUT 语句可以含两个以上的变量。当输入多个变量时,相互间要用逗号分隔开。

示例　　(输入数值)

```
10 REM "INPUT1"
20 INPUT "A=";A
30 INPUT "B=";B
40 C=A+B : D=A-B : E=A * B
50 PRINT "A+B="C
60 PRINT "A-B="D
70 PRINT "A * B="E
80 END
RUN    ↙     显示
A=? □    (方框的闪烁光标)
从键盘输入 8    ↙     显示
B=? □        输入 4  ↙打印
A+B=12
A-B=4
A * B=32
OK
```

若把上述程序中的 80 行改为 '

```
80 GOTO 20
RUN ↙
A=? □    输入 12 ↙ 显示
B=? □    输入 7  ↙ 显示
A+B=19
A-B+5
A*B=84
A=? □    输入 36 ↙ 显示
B=? □    输入 13 ↙ 显示
A+B=49
A-B=23
A*B=468
A=? □
```

输入数据和数据运算可以一直进行下去。

可见使用 INPUT 语句可以很方便地对确定算式的许多组不同数据进行反复运算，如计算学生的平均成绩等。

示例：(输入字符串)

```
10 REM "INPUT2"
20 INPUT "A$ =";A$ ,B$
30 PRINT A$ +B$
40 GOTO 20
RUN ↙
A$ =? □    输入 F ↙
? □        输入 BASIC ↙
FBASIC
A$ =? □
```

字符串和数值也可以混合输入，但输入的变量类型要一一对应。若输入了类型不同的变量计算机会作出不同的处理。如若要求输入字符串而输入了数字则把数字当做字符串处理；若要求输入数值而输入了字符串则把字符串当做 0 处理。

示例 （输入数字、字符）。

```
10 REM "INPUT3"
20 INPUT"A$,A,B=";A$,A,B
30 A$=A$+A$ : C=A*B
40 PRINT A$ , C
50 GOTO 20
RUN ↙
A$,A,B=? □      输入 ABC,12,7↙
ABCABC     84
A$,A,B=? □      输入 12,12,7 ↙
1212         84     (把第一个 12 当做字符)
A$,A,B=? □      输入 P,Q,7 ↙
PP           0     (把第二个字符当做 0)
⋮
⋮
```

例 3.14 分段统计学生成绩

分段统计学生成绩，分不及格；60～69；70～79；80～89；90～99，100 六个档次。要求分别打印统计结果。

程序如下：

```
5 REM "No. 3—17"
10 A=B=C=D=E=F
20 INPUT X
30 IF X>100 THEN 100
40 IF X<60 THEN A=A+1 : GOTO 20
50 IF X>=60 AND X<70 THEN B=B+1 : GOTO 20
60 IF X>=70 AND X<80 THEN C=C+1 : GOTO 20
70 IF X>=80 AND X<90 THEN D=D+1 : GOTO 20
80 IF X>=90 AND X<100 THEN E=E+1 : GOTO 20
90 IF X=100 THEN F=F+1 : GOTO 20
100 PRINT"<60="A,"60～69="B,"70～79="C,"80～89="D,"90～
```

```
    99="E,"100="F
110 END
```

程序中加入 30 行的目的是当输入大于 100 的数时结束数据输入，打印统计结果。

程序 No. 3—17 运行后，显示"?"号后跟一闪烁光标，输入一个得分回车后，则另起一行出现"?"等待再次输入。直到输入大于 100 的数，立即打印各档成绩人数(读者可输入数据练习)。

若使用开关转移语句，例 3.14 的程序要简洁得多(见 No. 3—18)。

```
5 REM "No. 3—18"
10 A=B=C=D=E=F
20 INPUT X
30 IF X>100 THEN 110
40 ON X/10+1 GOTO 50,50,50,50,50,50,60,70,80,90,100
50 A=A+1 : GOTO 20
60 B=B+1 : GOTO 20
70 C=C+1 : GOTO 20
80 D=D+1 : GOTO 20
90 E=E+1 : GOTO 20
100 F=F+1 : GOTO 20
110 PRINT "<60="A,"60～69="B,"70～79="C,"80～89="D,"90～
    99="E,"100="F
120 END
```

程序中 40 行为开关语句。X/10+1 是为了使得分从 0～59 变化时，关系式的值为 1～6，均转 50 行记录得分低于 60 的人数。当得分为 60 时关系式的值为 7，则转 60 行。余者以此类推。

3.3.2 LINPUT

格式：LINPUT"提示信息";<字符变量>

类型：语句或命令

简写:LIN.

功用:从键盘上接收整行字符的输入,并把它们当做一个字符串赋给一个串变量。

注释:LINPUT 与 INPUT 的相同之点都是从键盘上输入数据资料。不同之处是 LINPUT 语句中的提示信息、标点符号都可当作资料赋值给变量。另外 LINPUT 语句执行时不再出现问号。

注意:使用 LINPUT 语句时,一次输入的字符不要超过 31 个。否则将出现出错信息提示? ST ERROR IN ×××告诉你文字超过限额。程序示例:

```
10 REM "LINPUT"
20 LINPUT "STRING=";A$
30 PRINT A$
40 GOTO 20
RUN  ↙
STRING=□     输入 325,":    ↙ 打印
STRING=325,":
STRING=□  ……
```

说明:LINPUT 是 LINE INPUT 的缩写,英文的意思是整行输入。由于它可以把标点符号(尤其是双引号)赋值给变量,所以在某些场合该语句将十分有用。

3.4 数值函数

3.4.1 绝对值函数

格式:ABS(X)

类型:函数

简写:AB.

功用:对 X 的数值取绝对值。

注释:ABS 是 absolute value(绝对值)一词的缩写。

X 可以是－32768～＋32767 之间的任意数值或者是任意变量，也可以是算术表达式。

示例：

```
PRINT ABS(－112)        ↙
112     (已去掉了前面的负号)
PRINT ABS(18/2－25)        ↙显示
16
```

3.4.2 符号判别函数

格式：SGN(X)

类型：函数

简写：SG.

功能：确定给定数 X 的正负。符号判别函数可用于确定数值的取值范围。

注释：SGN 是 Sign 一词的缩写，是符号的意思。

X 可以是－32768～32767 之间的任意数，可以是变量或表达式。当 X 值为正时 SGN(X)＝1；若 X＝0 则 SGN(0)＝0；若 X 值为负则 SGN(X)＝－1。

示例：

```
PRINT SGN(10),SGN(0),SGN(－10)↙
10     0     －1
```

3.4.3 随机数函数

格式：RND(X)

类型：函数

简写：RN.

功用：产生一个均匀分布的随机数。

随机数是事先无法预计的数，它由计算机根据给定的 X 值不同，通过某种算法产生一个小于 X 的数字。

注释：RND 取自于英文 random(随机)一词。语句中的 X 为 1

～32767 之间的任意数。

示例：

```
10 REM "RND"
20 FOR I=1 TO 10
30 PRINT RND(20)
40 NEXT
RUN      ↙
1        19       14       17
4        3        6        16
13       2
```

为指导小学生进行数学练习，特介绍两个实用程序。

例 3.15　一位数乘法练习程序。

```
5 REM "No. 3－19"
10 A=RUN(10) : B=RND(10)
20 PRINT A"X"B"=";
30 INPUT C
"40 IF C=A * B THEN PRINT "vV" : GOTO 10
50 PRINT "vX" : GOTO 20
```

程序中"v"表示空格。

运行程序后，显示一位数乘法算式。输入得数后，若正确打印"V"，给出另一算式；若错误则打印"X"，重复打印算式……

```
RUN    ↙
2×3=? □        输入 6    ↙
V
5×8=?          输入  13  ↙
X
5×8=?          输入  40  ↙
V……
```

例 3.16　两位数加法练习程序

```
5 REM "No. 3－20"
```

```
10 A=RND(100) : B=RND(100)
20 PRINT A"+" B "=";
30 INPUT C
40 IF C=A+B THEN PRINT "vV" : GOTO 10
50 PRINT"vX" : GOTO 20
```

不难看出程序 No. 3－19、No. 3－20 格式是相同的，不同的是更换了随机数取值范围和运算符号。因此，读者可照此方法编写其它位数的加、减、乘练习程序。

3.5 数据语句

数据语句包括读数语句(READ)，置数语句(DATA)、恢复数据区语句(RESTORE)、存数语句(POKE)和取数语句(PEEK)

3.5.1 读数、置数语句(READ、DATA)

格式：READ ＜变量 1＞,＜变量 2＞,……

DATA ＜常数 1＞,＜常数 2＞,……

类型：语句

简写：REA.　D.

功用：READ 语句从 DATA 语句中读取数据，并把它赋给相应的变量。

DATA 语句为 READ 语句中的变量提供数据。

READ、DATA 语句适合于在对较多的变量赋值的场合下使用。DATA 语句的特点是一次可以提供一批数据。

注释：READ 是读的意思；DATA 是数据的意思。

当有许多变量需要赋值时，用 LET 语句就显得很繁琐。如有 3～9 七个自然数要分别赋值给 A～G 七个变量，若使用 LET 语句则要：

```
10 A=3 : B=4 : C=5 : D=6
```

```
20 E=7 : F=8 : G=9
```

若采用 READ 和 DATA 语句向变量提供数据则有：

```
10 READ A,B,C,D,E,F,G
20 DATA 3,4,5,6,7,8,9
```

当执行 READ 语句时，变量的值依次从 DATA 语句中读出。即 A 读 3、B 读 4、C 读 5……。

可以发现，当变量赋值较多时，用 READ、DATA 语句比较简单。

在使用 READ、DATA 语句时要注意两点：

1. 由于 DATA 是非执行语句，所以它可以放在程序中任意位置，但若有多个 DATA 语句时，它们之间必须有先后顺序。另外 DATA 语句提供的数据个数不能少于 READ 中变量的个数。如：

```
10 DATA 3,4,5,6
20 READ A,B,C
30 PRINT A,B,C
RUN      ↙
3          4          5
```

再如：

```
10 READ A,B,C
20 PRINT A,B,C
30 DATA 3,4
40 DATA 5,6
RUN      ↙
3          4          5
```

两程序的效果是一样的。

2. DATA 语句中的数据可以是数值、字符串，但不能是变量、函数或表达式。DATA 语句中的数据类型应严格与 READ 语句中的变量类型一一对应。如：

```
10 READ A$,AB,CDE
```

```
20 DATA "zhong Guo",1992,8,"comp"
30 PRINT A$,AB,CDE
```

READ 语句中有三个变量，第一个是串变量，后两个是数值变量。DATA 语句中有四个数据：1、4 项为字符串，2、3 项为数值。两语句中一、二、三项数据和变量类型分别对应，所以程序正确。至于 DATA 语句中多了一个串变量"comp"则没有关系，不影响程序的运行。

```
RUN        ↙
zhong guo 1992 8
```

另外，以后的叙述中将会看到 READ 语句还可以给下标变量赋值。

学习了置数、读数语句后，例 2.14 分类统计学生成绩的程序，可不必使用 INPUT 语句逐个输入学生成绩，而使用 DATA 语句装入学生成绩。修改程序 No. 3－18 如下：

```
20 READ X
120 DATA (学生成绩单),110
```

最后置 110 表示数据结束。

3.5.2 恢复数据区语句(RESTORE)

格式：RESTORE ＜行号＞

类型：语句

简写：RES.

功用：将数据区的指针恢复到某个 DATA 语句上。

READ 语句的变量每读取一项数据，DATA 数据表中的数据指针便向下移动一个数据项位置。当 DATA 数据表中的数据均被读完之后，指针便指到数据区的末尾。如果再次读取数据，指针就需要再恢复到前面某个数据项位置上，否则将出错。而 RESTORE 为我们反复使用同一批数据资源提供了方便。

注释：RESTORE 为恢复的意思。

语句中行号代表某个 DATA 语句。如果省略行号，则数据指针恢复到整个数据区首项数据位置上，即下一个 READ 语句从整个程序的第一个 DATA 语句的第一项数据开始重读数据。如果选用行号，则数据指针便恢复到该行号所指定的 DATA 语句的首项数据位置。

示例：如果要把 3～9 七个自然数分别送给变量 A～G、H～N、O～U，一般程序为：

```
10 READ A,B,C,D,E,F,G
20 READ H,I,J,K,L,M,N
30 READ O,P,Q,R,S,T,U
⋮
100 DATA 3,4,5,6,7,8,9
110 DATA 3,4,5,6,7,8,9
120 DATA 3,4,5,6,7,8,9
```

可见三行 DATA 语句是重复的。为节省内存简化程序，可使用 RESTORE 语句。当程序执行到 RESTORE 语句时，便使下一个 READ 语句需要的数据又从最小行号的 DATA 语句的第一个数据读起。于是上面的程序可改写为：

```
10 READ A,B,C,D,E,F,G
20 RESTORE : READ H,I,J,K,L,M,N
30 RESTORE : READ O,P,Q,R,S,T,U
⋮
100 DATA 3,4,5,6,7,8,9
```

我们在介绍开关转移语句时，曾讲过四种格式。下面介绍 ON RESTORE 格式的使用。

```
10 REM "ON RESTORE"
20 A=0
30 A=A+1 : IF A>4 THEN END
40 ON A RESTORE 130,120,110,100
```

```
50 READ B,C,D,E,F,G
60 PRINT "A=" A
70 PRINT B,C,D,E,F,G
80 GOTO 30
100 DATA 1,2,3,4,5,6
110 DATA 11,12,13,14,15,16
120 DATA 21,22,23,24,25,26
130 DATA 31,32,33,34,35,36
RUN        ↙
A=1
31          32          33          34
35          36
A=2
21          22          23          24
25          26
A=3
11          12          13          14
15          16
A=4
1           2           3           4
5           6
```

可见使用 ON—RESTORE 语句方便地控制了数据区指针的改变。

3.5.3 三种提供数据语句的比较

现在我们已经学习了 LET、INPUT、READ/DATA 三种语句。这三种语句都可以给变量赋值，同一个问题可分别用三种不同的方法编写程序。现举例说明：

例 3.17 鸡兔同笼，已知鸡兔总头数为 T(TOU)，总脚数为 J(Jiao)，求鸡兔各有多少只？

先用代数求出鸡兔只数的数学表达式。

设　　鸡有 X 只，兔有 Y 只。则有

$$\begin{cases} X=(4T-J)/2 \\ Y=(J-2T)/2 \end{cases}$$

今设　T＝16，J＝40，求 X、Y，编程如下：

程序 I：用 LET 语句

```
5 REM "No. 3－21"
10 T=16:J=40
20 X=(4*T－J)/2
30 Y=(J－2*T)/2
40 PRINT "Ji="X,"Tu="Y
```

程序 Ⅱ：用 INPUT 语句

```
5 REM "No. 3－22"
10 INPUT "T,J=";T,J
20 X=(4*T－J)/2
30 Y=(J－2*T)/2
40 PRINT "Ji="X,"Tu="Y
```

程序 Ⅲ：用 READ/DATA 语句

```
5 REM "No. 3－23"
10 READ T,J
20 X=(4*T－J)/2
30 Y=(J－2*T)/2
40 PRINT "Ji="X,"Tu="Y
50 DATA 16,40
```

由上例可见，三种语句的相同之处是都能提供原始数据，但使用特点各有不同。

LET 语句适合于赋值变量少，且编程时数据已确定的场合。赋值语句还可用于计算，这一功能，其它两种无法取代。

如果数据很多，且编程时已确定，使用 READ/DATA 语句比较方便。如统计学生成绩、排列名次等。

如参数是变化的，要随时输入变化了的数据，则宜于采用 IN-

PUT 语句。如商业柜台、银行付息的计算等。

三种语句掌握熟练后，定会给编程带来极大的方便。希读者多加练习。

3.5.4 对内存单元置数、读数语句

在程序设计中往往要与内存单元交换信息，这就要使用向内存单元置数语句(POKE)和从内存单元读数语句(PEEK)。

格式：POKE <地址>,<数据>,……

PEEK (地址)

类型：语句

简写：PO.　PE.

功用：POKE 的功用是将某些字节值(0～255)存入某一地址单元。

PEEK 的功用是查询内存储器中某一地址单元中存储的内容。

注释：POKE 语句的是具体功能是将数据放入地址所指定的内存单元内。地址可以取常数、变量或表达式，其取值范围为－32768～32767。数据是一个取值在 0～255 之间的整数，也可取变量形式，该值作为存储内容送入地址所指定的单元内。

PEEK 语句中的地址代表被检查的内存单元。它是一个取值在－32768～32767 之间的整数或表达式。也可以取变量形式。PEEK 所返回的是该地址所指定的存储单元中的内容，这个数值在 0～255 范围内。

注：在有些资料中把 PEEK 语句作为函数对待。一个 POKE 语可以写入多个数据，系统会令地址自动加一。

示例：

```
10 REM "POKE--PEEK"
20 FOR I=0 TO 100
30 PRINT PEEK(30208+I),
```

```
40 NEXT
50 FOR I=0 TO 100
60 POKE 30208+I,I
70 NEXT
80 PRINT "OUT AGAIN"
90 FOR I=0 TO 100
100 PRINT PEEK(30208+I),
110 NEXT
```

程序运行后首先打印内存单元从30208～30308中的初值。然后打印"OUT AGAIN",打印0～100,表示对30208～30308内存写入了新值。

有时需要了解内存中某一块存储单元的内容,可在打印语句中加入打印内存地址的语句,使地址和内容一一对应。如查看地址为800～820内存中的内容,可编制如下程序:

```
10 REM"PEEK"
20 FOR I=800 TO 820
30 PRINT "("I")="PEEK(I),
40 NEXT
RUN
```

```
(800)=254    (801)=253    (802)=32
(803)=255    (804)=33     (805)=106
(806)=4      (807)=255    (808)=255
(809)=255    (810)=255    (811)=33
(812)=138    (813)=4      (814)=254
(815)=32     (816)=254    (817)=253
(818)=33     (819)=170    (820)=4
```

3.6 数组及数组说明语句

在第一章我们介绍了简单变量,下面介绍变量的第二种形式

——下标变量。

3.6.1 下标变量

顾名思义，下标变量就是带有下标的变量，如 A1、B32、C123 等。在计算机语言中，为与简单变量区别，下标一般用括号括起来，如 A(1)、B(32)、C(123)等。使用下标变量的目的是为了使一些性质相近的变量便于归类。如 A(K)而言，K＝1 时，A(K)就是 A(1)；K＝2 时，A(K)就是 A(2)……。

F BASIC 语言允许下标变量名包含两个下标，其形式为 A(I，J)。一般称一个下标的为单下标变量；两个下标的为双下标变量。

3.6.2 数组

由同一个变量名组成的下标变量的集合叫数组。在实际应用中，人们习惯把一些具有相同性质的数据放在一个数组内，用不同的下标去区分它们。

由单下标变量组成的数组称一维数组；由双下标变量组成的称二维数组。当数组进入电脑后每个下标变量或元素要占用两个内存单元，因此一个数组要占用内存中连续的一组单元。

3.6.3 数组说明语句(DIM)

数组在使用前要先定义。用一个数组说明语句 DIM 就可以定义数组，以规定数组的名称及包含的下标变量的个数。

格式：DIM ＜数组名 1(元素数目)＞，

＜数组名 2(元素数目)＞，……

类型：语句

简写：DI.

功用：建立或定义数组，即在内存中为数组开辟存储空间，确定数组名、数组数目、数组类型、数组维数及各维元素数目。

注释：DIM 源于 dimension，维数的意思。

数组名代表所要定义的数组，它与变量名类似。但数组与普通的数值变量或字符串变量又有区别，它里面存放的是若干个有序

元素，每个元素都是一个变量。这些元素或者均是数值型的，或者均是字符型。如果是字符型的，定义数组名时要像定义串变量一样带上$符号。DIM语句一次可只定义一个数组，也可以定义多个数组。

在对每个数组进行定义时，也要同时定义数组的维数。维数的多少是由DIM语句中的各数组名之后所跟元素数目项数的多少决定的，各个元素数目之间要用逗号分隔开。每个元素数目又是用来确定本维元素数目的，即本维所能容纳的元素个数(一般也是本维元素的最大下标号)。元素数目一般用一个数值常数表示，也可以用数值变量或表达表示。如A(13)，元素数目项数为1，所以它是一维数组；B(2,3)则是二维数组。同时A(13)也表示一维数组所能容纳的元素个数是14个，即最小下标变量是A(0)，最大下标变量是A(14)。可见对于一维数组A(m)，其变量个数为m+1。同理，对于二维数组A(m,n)，变量个数则为(m+1)*(n+1)个。

注意：

1. 数组必须先定义后使用，因此DIM语句一般放在程序的前面。

2. 在同一程序中，同一名称数组只能定义一次，除非使用CLEAR语句清除，否则不能反复定义。

3. 每维元素限定在255个以内。

例 3.18　从DATA语句中读出10个数据存于A数组中，然后将数据次序颠倒后再存于B数组中。

程序如下：

```
5 REM "No. 3－24"
10 DIM A(9),B(9)
20 DATA 3,4,5,6,7,8,9,10,11,12
30 FOR I=0 TO 9
40 READ A(I) : NEXT
```

```
50 RESTORE
60 FOR L=9 TO 0 STEP-1
70 READ B(L) : NEXT
80 FOR M=0 TO 9
90 PRINT A(M),B(M) : NEXT
RUN ↙
3          12
4          11
5          10
6          9
7          8
8          7
9          6
10         5
11         4
12         3
```

例 3.19 有 10 个杂乱无章的数，要求编一程序把这些数按大小顺序排列出来。

```
5 REM "No. 3-25"
10 DIM A(9)
20 DATA 38,99,26,78,32,100,27,66,95,46
30 FOR K=0 TO 9: READ A(K) : NEXT
40 FOR L=0 TO 8
50 FOR N=0 TO 8-L
60 IF A(N)<=A(N+1) THEN 80
70 A=A(N) : A(N)=A(N+1) : A(N+1)=A
80 NEXT : NEXT
90 FOR I=0 TO 9 : PRINT A(I),
100 NEXT
RUN ↙  打印按升序排列的一组数据。
26    27    32    38
```

```
46    66    78    95
99    100
```

若要求输出按降序排列则只须把 90 行改为：90 FOR I=9 TO 0 STEP−1 : PRINTA(I),RUN ↙ 则打印按降序排列的一组数据。

```
100    99    95    78
66     46    38    32
27     26
```

通过例 3.19 可以看出，使用 DIM 语句，可以对学生成绩排列名次。读者可试编一程序。

在例 3.14 分段统计学生成绩的程序中，我们使用了开关转移语句虽使程序变得简洁（见程序 No. 3−18），但总觉得 50～100 行语句重复得太多。若引入下标变量代替 No. 3−18 中的简单变量将使程序极大地简化，见程序 No. 3−26。

```
5 REM "No. 3−26"
10 DIM A(5)
20 INPUT X : IF X>100 THEN 50
30 N=X/10 : IF N<6 THEN A(0)=A(0)+1 : GOTO 20
40 A(N−5)=A(N−5)+1 : GOTO 20
50 PRINT "<60=" A(0),"60～69="A(1),"70～79="A(2),"80～89
   ="A(3),"90～99="A(4),"100="A(5)
60 END
```

请读者运行程序，输入数据验证，No. 2−26 与 No. 2−18 的功能是一样的，但程序结构要简单多了。这也进一步体现了编程的灵活性。

3.7 交换变量内容语句(SWAP)

在例 3.19 的排序程序(No. 3−25)70 行的语句中，为交换 A(N)、A(N+1)两下标变量中的内容，我们引进了中间变量 A。使 A

记录A(N)中的内容，再把A(N+1)中的内容送给A(N)，最后把A中的内容送给A(N+1)，从而实现内容的交换。其实交换变量的内容还有更简便、更直接的方法，就是使用SWAP语句。

格式：SWAP <变量1>，<变量2>

类型：语句

简写：SW.

功用：交换两个变量的值。

利用这一语句可以对两个或两个以上的变量进行比较或对多个数据进行排序。

注释：SWAP为交换或者对调的意思。

变量1和变量2是其值将要被交换的两个变量或数组元素。该语句执行后，前者的原值便会送给后者，后者的原值也会送给前者。

变量1和变量2在类型上要保持一致，或者均为串型，或者均为数值型。

示例：

```
10 REM "SWAP1"
20 A$ ="ABC" : B$ ="XYZ"
30 PRINT "START ";"A$ ="A$ ,"B$ =" B$
40 SWAP A$ ,B$
50 PRINT "END       ";"A$ ="A$ ,"B$ ="B$
RUN        ↙
START   A$ =ABC          B$ =XYZ
END     A$ =XYZ          B$ =ABC
10 REM "SWAP2"
20 DIM A(4)
30 FOR X=1 TO 4 : A(X)=X
40 PRINT "A(" X ")=" A(X), : NEXT
50 PRINT
```

```
60 SWAP A(1),A(4)
70 FOR X=1 TO 4
80 PRINT "A(" X ")=" A(X), : NEXT
90 END
RUN ↙
A(1)=1    A(2)=2    A(3)=3    A(4)=4
A(1)=4    A(2)=2    A(3)=3    A(4)=1
```

同样程序 No. 3—25 中的 70 行可改写为：

```
70 SWAP A(N),A(N+1)
```

可见使用 SWAP 语句交换变量的内容极为方便。

3.8 清除语句（CLEAR）

清除语句 CLEAR 有两个格式、两种功能，下面分别介绍：

格式：CLEAR

类型：语句、命令

简写：CLE.

功用：将内存中的所有数值变量清零，将所有字符串变量清空。取消已定义的数组。

注释：CLEAR 为清除或清零的意思。

示例：

```
10 REM "CLEAR"
20 A+10 : B=3000 : C=15625
30 A$ ="USE" : B$ ="COMPUTER"
40 PRINT "A="A,"B="B,"C="C
50 PRINT "A$ ="A$ ,"B$ ="B$
60 CLEAR
70 PRINT "USED CLEAR"
80 PRINT "A="A,"B="B,"C="C
```

```
90 PRINT "A$="A$,"B$="B$
100 END
RUN
A=10        B=3000        C=15625
A$=USE    B$=COMPUTER
USE CLEAR
A=0         B=0           C=0
A$=         B$=
```

运行可见，通过 60 行 CLEAR 的作用，使数值变量 A、B、C 全部清零；串变量 A$、B$ 全部清空。

格式 2：CLEAR ＜数值＞

类型：语句、命令

简写：CLE. ＜数值＞

功用：规定用户使用的内存空间范围。

为了防止因用户程序太长而与 BASIC 系统内存空间相冲突，使用 CLEAR 语句规定用户程序在内存中存放的起始位置。如

CLEAR &H7600

规定程序不能用十六进制 7600 地址以前的内存空间，只能从 7600 开始向后存放。

注释：语句中的数值取值范围：十六进数为 &H0～&HFFFF。

注意：该功用只有在对话进入 BASIC 时有效，而在直接进入 BASIC 时无效。

3.9 程序运行控制语句

程序运行控制语句有三条：STOP、CONT 和 PAUSE。

3.9.1 中断语句(STOP)

格式：STOP

类型：语句

简写:STO.

功用:中断程序运行,将计算机置于 BASIC 的命令方式或立即方式。

中断运行的目的是为了调试程序,观察某些变量在程序运行期间的中间结果。

注释:STOP 是停止或中断的意思。

STOP 可以放在程序中的任何语句之后,程序运行中断之后,计算机会以下述方式显示中断时 STOP 语句所在行号:

BREAK IN ×××　　(×××表示中断的行号)

此时,用户可以修改程序。然后用 CONT 指令恢复程序运行。

示例:

```
10 REM "STOP"
20 FOR X=1 TO 100
30 IF X=10 THEN STOP
40 PRINT X,
50 NEXT
60 END
RUN ↙
1          2          3          4
5          6          7          8
9
BREAK IN 30
```

键入 CONT 命令则继续打印

10　　11　　12　　13

14……

注意:程序运行中断后,若对程序进行了修改,则 CONT 命令无法恢复运行。

3.9.2 解除中断命令(CONT)

格式:CONT

类型:命令

简写:C.

功用:使中断后的程序恢复运行。

注释:CONT 取自于 Continue,为继续的意思。CONT 只适用于在程序运行中断后使用。运行中断后 BASIC 将给出断点位置(行号)。

3.9.3 延时语句(PAUSE)

格式:PAUSE <数值>

类型:语句

简写:PA.

功用:令程序运行暂停一段时间后再继续运行。

PAUSE 语句,主要用于程序运行中的延时或暂停。

注释:PAUSE 是"中止或暂停"的意思。

PAUSE 语句中的数值为暂停时间或称为延时量。取值范围为 0～32767。

示例

```
10 REM "PAUSE 1"
20 FOR I=0 TO 10
30 PAUSE 30
40 PRINT I * I,
50 NEXT
RUN    ↙  分别延时一段时间打印一个数字。
0       1       4       9
16      25      36      49
64      81      100
```

当 PAUSE 后不带数值时,则程序暂停,出现一闪烁光标,等待用户从键盘上输入任一字符。演示程序如下:

```
10 REM "PAUSE 2"
20 PRINT "PAUSE"
```

```
30 PRINT "  PRESS ANY KEY TO CONTINUE…"
40 FOR I=0 TO 10
50 PAUSE
60 PRINT I * I
70 NEXT
RUN  ↙
PAUSE
PRESS ANY KEY TO CONTINUE…
```

这时出现闪烁光标，等待用户按任一键，每按一键，打印一个数字。连续按键则打印出 11 个数字，结果同上。

3.10 子程序调用语句

子程序调用语句由转子(转移到子程序)语句 GOSUB 和返回语句 RETURN 组成。

在一个程序中，某些部分往往要反复执行，为避免重复编写实现同一目的的程序段而引入了子程序。子程序为一具有某种功能的程序，它是独立的，可以放在主程序之外。当主程序需要这一功能时，就可调用这一子程序。

3.10.1 子程序的调用

主程序调用子程序的方法很简单，即编一个转子语句，就可以转去执行子程序。转子语句的格式如下：

格式：GOSUB <行号>

类型：语句

简写：GOS.

功用：将程序控制权交给指定的子程序。

子程序可以完成程序的某一部分功能。

采用转子方法可使程序的结构和层次清晰，便于阅读和修改。

注释：GOSUB 取自于 GO to a subroutine，是“转去执行子程序”

的意思。

行号为转移的目标，即子程序的首行语句标号，它代表指定的子程序。行号要取数值常数形式，取值范围为 0～65535。

子程序的最后一行语句必须是返回语句。返回语句的格式如下：

格式：RETURN <行号>

类型：语句

简写：RE.

功用：RETURN 通常用在子程序的末尾，其功能是结束子程序的运行，将控制权交还给调用程序。

RETURN 必须与 GOSUB 语句联用，否则程序执行到 RETURN 时要给出错误信息提示 RG ERROR IN×××

并中断程序运行。

注释：RETURN 为返回的意思。

语句中行号为可选项，用于确定返回的位置。如果省略则返回到调用程序中 GOSUB 的下一语句。

示例：

在第 3.8 节介绍清除语句 CLEAR 时，我们曾列举了示例程序。在该程序中 40、50 行和 80、90 行完全相同，显得整个程序臃肿。我们可使用转子语句使程序简洁。示例如下：

```
10 REM "GOSUB—RETURN"
20 A=10 : B=3000 : C=15625
30 A$ ="USE" : B$ ="COMPUTER"
40 GOSUB 100
50 CLEAR : PRINT "USED CLEAR"
60 GOSUB 100 : END
100 PRINT "A="A, "B="B, "C="C
110 PRINT "A$ ="A$ , "B$ ="B$
120 RETURN
```

读者可运行程序检验，两程序功效是一样的。

例 3.20　设 A＝4，B＝5，C＝6，试编一程序求 S＝A！＋B！＋C！。

程序如下：

```
5 REM "No. 3－27"
10 GOSUB 100
20 A=P
30 GOSUB 100
40 B=P
50 GOSUB 100
60 C=P
70 PRINT "="；A+B+C：END
80 DATA 4,5,6
100 P=1：READ M
110 FOR J=1 TO M：P=P*J：NEXT
120 PRINT "+"M"!"；
130 RETURN
RUN ↙
+4! +5! +6! =864
```

为便于理解，程序 No. 3－27 书写的比较烦琐。读者在弄通编程思路后，可把 No. 3－27 简写如下：

```
5 REM:No. 3－28
10 GOSUB 100：GOSUB 100：GOSUB 100
20 PRINT "="; A：END
30 DATA 4,5,6
100 P=1：READ M
110 FOR J=1 TO M：P=P*J：NEXT
120 A=A+P：PRINT "+" M "!";
130 RETURN
```

可见 No. 3－28 简洁多了。但是 No. 3－28 中的 10 行仍嫌罗嗦，可改为：

```
10 FOR I=1 TO 3 : GOSUB 100 : NEXT
```

改动后，程序运行结果一样。请读者验证。

3.10.2 开关转子语句(ON—GOSUB)

ON—GOSUB 语句是开关转移语句的第二种形式。

格式：ON ＜表达式＞ GOTO ＜行号 1＞，＜行号 2＞，＜行号 3＞，……

类型：语句

简写：O.　GOS.

功用：根据表达式取值大小从多个分支中选择并执行其中某一个分支。ON—GOSUB 也称为多子选择转向语句。

注释：ON—GOSUB 是“当……时则转去执行……子程序”的意思。

表达式可以是数值变量，也可以是数值表达式，其数值取值范围为 1～255，其值的大小决定着转移的方向。

行号是指各个目标子程序的起始行号，代表各个备选的子程序。

该语句按变量取值情况决定转移目标。若表达式的值等于 1，则执行行号 1 所代表的子程序；如表达式的值等于 2，则执行行号 2 所代表的子程序。以此类推。就是说，表达式的值与行号位置是一一对应的。当表达式的值为 0 或大于 255 时，则 BASIC 顺序执行 ON—GOSUG 的下一语句。

示例

```
10 REM“ON—GOSUB”
20 A=A+1 : IF A=10 THEN END
30 ON RND(4)+1 GOSUB 100,200,300,400
40 GOTO 20
100 PRINT “A=”A,“AAAA”
110 RETURN
200 PRINT “A=”A,“BBBB”
```

```
210 RETURN
300 PRINT "A="A,"CCCC"
310 RETURN
400 PRINT "A="A,"DDDD"
410 RETURN
RUN     ↙     可能打印
A=1       AAAA
A=2       DDDD
A=3       AAAA
A=4       CCCC
⋮
⋮
A=9       BBBB
```

每次运次打印的结果各不相同。

3.10.3 子程序的嵌套

子程序也可以再调用其它子程序，这叫做子程序的嵌套，但是不允许子程序调用主程序。F BASIC 系统允许子程序调用它自己。

对于例 3.20 求阶乘之和的问题也可以使用子程序嵌套的方法解决。编程如下。

```
5 REM "No. 3-29"
10 FOR I=1 TO 3 : GOSUB 100 : NEXT
20 PRINT "="A : END
30 DATA 4,5,6
100 P=1 : READ M : GOSUB 200
110 RETURN
200 FOR J=1 TO M : P=P*J : NEXT
210 GOSUB 300 : RETURN
300 A=A+P : PRINT "+" M "!"; : RETURN
```

程序中使用了子程序的三重嵌套。对于解决例 3.20 的问题，读者可能觉得 200、300 两个子程序有点多余。但是当在一个较大

的程序中，其它程序还要调用 200、300 时将显示出这种编程方法的优越性。

关于子程序嵌套的使用，我们将于本书的下篇，编程技巧部分详细介绍。

3.10.4 开关返回语句

在发生子程序调用时 BASIC 执行完子程序后要返回调用程序，一般返回调用语句的下一语句。但有时出于某些特殊需要，要根据不同情况返回调用程序的不同部分，欲达到这一目的必须使用开关返回语句 ON－RETURN。这一语句是 F BASIC 语言所特有的。它相当于多个条件转移语句的缩写。

格式：ON ＜表达式＞ RETURN ＜行号 1＞，＜行号 2＞，＜行号 3＞，……

类型：语句

简写：O. RE.

功用：根据表达式取值大小从多个返回调用程序分支中执行其中某一个分支。ON－RETURN 也称为多项选择返主语句。

注释：ON－RETURN 是“当……时则返回……主程序”的意思。

表达式可以是数值变量，也可以是数值表达式，其数值取值范围为 1～255，其值的大小决定着返回调用程序的方向。

行号是指返回调用程序的行号。

示例

```
10 REM "ON－RETURN"
20 PRINT : INPUT "A=";A
30 GOSUB 500
40 PRINT "A>4,INPUT AGAIN!" : GOTO 20
100 PRINT "A=1, DO 100" : GOTO 20
200 PRINT "A=2, DO 200" : GOTO 20
```

```
300 PRINT "A=3, DO 300" : GOTO 20
400 PRINT "A=4, DO 400" : GOTO 20
500 ON A RETURN 100,200,300,400
510 RETURN
RUN    ↙      显示
A=?  □        键入 1    ↙
A=1,DO 100(表示 A 等于 1 执行 100 行。)
A=?  □        键入 3    ↙
A=3,DO 300    (A 等于 3 执行 300 行。)
A=?  □        键入 5    ↙
A>4,INPUT AGAIN! (A 大于 4,重新输入。)
A=4  □        键入 4    ↙
A=4,  DO  400
⋮
```

BASIC 根据 A 的取值返回不同的程序行。

3.10.5 子程序的递归调用

一个子程序中,如果包含对自己的调用,则称为递归调用。递归是解决许多带有回溯性质的算法问题的有效方法。同时递归的应用还使得程序具有良好的结构和可读性。

示例

```
5 REM "GOSUB—GOSUB"
10 GOSUB 100
20 PRINT "AAAA"
30 END
100 PRINT "BBBB"
110 A=A+1 : IF A=4 THEN RETURN
120 GUSUB 100
130 RETURN
RUN    ↙          打印
BBBB
```

BBBB

BBBB

BBBB

AAAA

程序中 120 行实现了对 100 行子程序的递归调用。递归调用 3 次，通过 110 行返回主程序。

为进一步说明递归调用的应用，再举一应用程序实例。

例 3.21　试编程找出下列字母方阵(图 3－2)中所有隐藏的单词 F BASIC，单词中字母的走向可以是图 3－3 中编号为 1—8 的八个方向，要求打印格式为：

(X,Y),a1,a2,a3,a4,a5

(X,Y)为首字母“F”所在的行、列座标，a1～a5 是后续每个字母相对前一个字母的走向代号。例如图 3－2 中箭头所示的 F BASIC 是其中的一种，其打印格式为：

(1,1),3,5,4,6,8

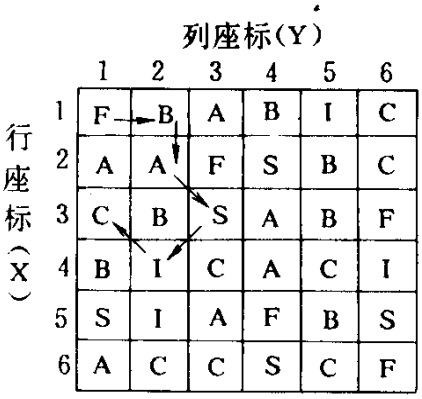


图 3－2

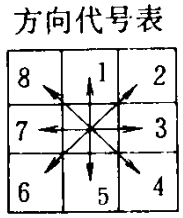


图 3－3

程序如下：

```
5 REM "No. 3－29"
10 F$ ="F BASIC" : N=LEN(F$) : DIM H$(6,6),A(8),B(8),X
   (N),Y(N),T(N)
20 FOR I=1 TO 6 : FOR J=1 TO 6 : READ H$(I,J) : NEXT : NEXT
```

```
30 FOR S=1 TO 8 : READ A(I),B(I) : NEXT
40 FOR P=1 TO 6 : FOR J=1 TO 6 : X(1)=P : Y(1)=J
50 IF H$(X(1),Y(1))="F" THEN K=2 : T(2)=0 : GOSUB 500
60 NEXT : NEXT : END
70 DATA F,B,A,B,I,C : DATA A,A,F,S,B,C
80 DATA C,B,S,A,B,F : DATA B,I,C,A,C,I
90 DATA S,I,A,F,B,S : DATA A,C,C,S,C,F
100 DATA -1,0,-1,1,0,1,1,1,1,0,1,-1,0,-1,-1,-1
500 REM "Zi Cheng Xu"
510 T(K)=T(K)+1 : IF T(K)>8 THEN 580
520 X(K)=X(K-1)+A(T(K)) : Y(K)=Y(K-1)+B(T(K))
530 IF X(K)<1 OR X(K)>6 OR Y(K)<1 OR Y(K)>6 THEN 510
540 IF H$(X(K),Y(K))<>MID$(F$,K,1) THEN 510
550 IF K<N THEN K=K+1 : T(K)=0 : GOSUB 500 : K=K-1 : GO-
    TO 510
560 PRINT "("X(1)","Y(1)")"; : FOR I=2 TO N : PRINT ","T(I); :
    NEXT
570 PRINT : GOTO 510
580 RETURN
```

RUN ↙ 顺序打印结果如下

```
(1,1),3,3,4,2,3
(1,1),3,3,4,2,4
(1,1),3,5,4,6,3
(1,1),3,5,4,6,8
(2,3),2,7,4,2,3
(2,3),2,7,4,2,4
(2,3),6,1,4,6,3
(2,3),6,1,4,6,8
(2,3),8,3,4,2,3
(2,3),8,3,4,2,4
(2,3),8,5,4,6,3
```

(2,3),8,5,4,6,8
(3,6),7,6,8,6,3
(3,6),7,6,8,6,8
(3,6),7,7,1,2,3
(3,6),7,7,1,2,4
(3,6),7,7,7,6,3
(3,6),7,7,7,6,8
(3,6),8,6,1,2,3
(3,6),8,6,1,2,4
(3,6),8,6,7,6,3
(3,6),8,6,7,6,8
(5,4),3,8,8,6,3
(5,4),3,8,8,6,8
(6,6),8,8,8,6,3
(6,6),8,8,8,6,8

读者可以验证一下，图 3—2 中的字母走向是否只有这 26 种。

程序 No. 3—29 中使用了字符串语句 MID $，关于这一语句的作用，我们将在后续章节介绍。本程序比较难懂，读者也不要强求读通。在下篇编程技巧中我们将详细讨论递归调用问题。

我们提前列出这一程序，主要是为了使读者一窥子程序递归调用的奇妙。No. 3—29 仅用了 18 条语句就解决了这一复杂的问题。若不使用递归调用则程序量要增大好几倍。

注：例 3.21 为福建省 1988 年青少年计算机程序设计竞赛题。原文发表在《电子与电脑》1990 年第三期。作者对其改造后引入本文。

3.11 程序调试命令

在程序的编写和调试过程中常用的指令有自动提供行号命令

(AUTO)、删除行号命令(DELETE)、重编行号命令(RENUM)和查找命令(FIND)。这四条命令在键盘的随机手册中没有介绍,但在程序的编写、调试中用途较大,下面分别介绍。

3.11.1 自动行号命令(AUTO)

格式:AUTO <起始行号>,<增量>

类型:命令

简写:A.

功用:使 BASIC 能够在用户编写程序时,在每一语句键入之前自动提供语句标号。

使用 AUTO 命令后,用户不必每书写一条语句都用手工办法来输入行号,只输入语句本身即可。从而有助于简化编程工作,减少失误和差错,提高编程效率。

注释:AUTO 源于 Auto matic,在英文中为自动的意思。

起始行号代表自动编号时的起始语句行,它的取值范围在 0～65535 之间。若省略起始行号,则 BASIC 默认其值为 10。

增量是指自动编号时两个相邻语句行行号的间隔。如果该项省略,则 BASIC 假定增量为 10。

如果起始行号和增量都省略,则 BASIC 默认起始行号为 10、增量为 10。

AUTO 命令既可以在编程开始时使用,也可以在编程过程中间使用。该命令一经输入,BASIC 就会自动产生一个由起始行号所指定的语句行号,这时用户可在该行号之后输入一个语句。当写完一个语句并按下回车键后,BASIC 便又会产生下一语句的行号。行号的具体值等于上一语句行号加上增量。

示例:下面给出几种使用 AUTO 命令的典型例子:

命　令	产生的行号序列
AUTO	10,20,30,40……
AUTO 5	5,15,25,35……

AUTO 10,20　　10,30,50,70……

AUTO 8,8　　8,16,24,32……

AUTO ，100　　0,100,200,300……

说明：

1. 如果在 AUTO 命令前内存中已经驻留有一些 BASIC 程序语句，当自动编的行号同现有程序行号相重时，则会显示该行号的语句。可修改本行语句，也可不修改，按回车键则跳过该行，显示下一行。

2. 自动编行号过程一直到按下 Ctrl＋C 键或 STOP 键才告结束。结束后也可再次使用 AUTO 命令。再次使用 AUTO 时，如果省略了增量将会取上次 AUTO 所设的间隔值。

3.11.2 重编行号命令(RENUM)

格式：RENUM ＜新行号＞,＜现行号＞,＜增量＞

类型：命令

简写：REN.

功用：重编程序语句行号。

当程序修改之后，语句行号将会变得很不整齐，甚至会出现相邻两语句行号过近而无法插入新语句的现象。使用 RENUM 命令重新编号，不仅可使程序清楚易懂，而且也便于插入一些新的语句。

RENUM 命令发出后，程序中 GOTO、GOSUB、IF－THEN、ON－GOTO、ON－GOSUB 等转移语句的目标行号也将会随之发生变化。

注释：RENUM 取自于 Renumber，为重新编写行号的意思。

如果 RENUM 后各项全部省略，则无论程序原来的行号怎样，RENUM 均按起始行号为 10、增量为 10 的规则重新对程序的行号进行整理，使其变为 10、20、30……的形式。

如果仅选用新行号，则整理后的程序将会以新行号为第 1 行

语句的行号。

如果选用现行号，则 RENUM 将从现行号所指定的语句开始整理程序，其前面的语句行号不变，现行号所代表的那条语句行号会改成新行号所指定的值，其后的各个语句的行号将以指定的增量为间隔。

示例：

RENUM

这一命令会把程序行号整理成：10，20，30……。

RENUM 1000，180

这个命令可对第 180 行以后的语句重新进行编号。第 180 行语句以前的行号不变。180 行变为 1000，随后的各行为 1010，1020，……。

```
10 FOR A=1 TO 55
18 B=A * A
22 IF B>35 GOTO 33
25 PRINT A,
31 NEXT
33 PRINT "B>35"
RENUM
LIST
10 FOR A=1 TO 55
20 B=A * A
30 IF B>35 GOTO 60
40 PRINT A
50 NEXT
60 PRINT "B>35"
```

可见，使用 RENUM 命令后杂乱的行号变得整齐划一。

3.11.3 删除命令(DELETE)

在调试程序中，有时要删除部分语句行，这时可使用 DELETE

命令达到目的。

格式:DELETE <行号 1>—<行号 2>

类型:命令

简写:DEL.

功用:删除部分程序语句。

在编辑程序过程中,如果想一次删除一段程序,使用 DELETE 命令是很方便的。

注释:DELETE 为删除的意思。

行号 1、行号 2 分别代表被删除程序段的首行语句和末行语句,其作用是确定被删除的程序段。两个行号可以省略一个,如果省略前者,则 DELETE 命令删除自程序第一条语句开始到行号 2 为止的全部语句;如果省略后者,则删除自行号 1 开始到程序最后一条语句为止的全部语句。

若只删除一条语句,可打入该语句行号回车就可删除,也可以用 DELETE <行号>删除。

示例:下面列出 DELETE 命令应用的几个典型例子:

命　令	结　果
DELETE 100	删除程序的第 100 行语句。
DELETE 50—90	删除第 50～90 行语句。
DELETE 40—	删除第 40 行及以后的所有语句。
DELETE —200	删除第 200 行及之前的所有语句。

3.11.4 查找命令(FIND)

在调试程序中,若欲查找某一数字、某一变量、某一表达式或字符串在程序中的位置,可使用 FIND 命令。该命令是 F BASIC 特有的命令。

格式:FIND <"查找内容">

类型:命令

简写:FI.

功用：列出查找内容在程序中的语句。

FIND 命令在阅读别人编制的程序时尤其方便重要。

注释：FIND 为寻找的意思。

查找内容可以是数值、变量、字符串、表达式或者语句。但须注意所查找的内容必须用双引号括起来。

示例

例 3.22　民间趣题　　韩信点兵。

设某数 N 被 A 除余数为 X，被 B 除余数是 Y，被 C 除余数是 Z，试编程求 N。

根据题意有

$$\begin{cases} N \text{ MOD } A=X \\ N \text{ MOD } B=Y \\ N \text{ MOD } C=Z \end{cases}$$

程序如下：

```
5 REM "No. 3-30"
10 INPUT A,B,C
20 INPUT X,Y,Z
30 D=A * * C
40 FOR N=1 TO D
50 IF N MOD A=X THEN IF N MOD B=Y THEN IF N MOD C=Z THEN
   70
60 NEXT
70 PRINT N; GOTO 20
RUN    ↙      显示提示光标
```

先输入除数，如：除数分别为 3，5，7，则键入

```
3,5,7      ↙  显示闪烁光标
```

再输入余数，如 N=95 则有

```
95 MOD 3=2
95 MOD 5=0
```

95 MOD 7=4

则键入

2,0,4　　↙　则输出结果

95

如欲改变除数，如：除数分别为 5,9,13，则键入 STOP 中断程序运行，再键入 RUN 重新运行程序，依次输入“5,9,13”，再依次输入余数，就可打印出结果。

“韩信点兵”这一民间趣题原意仅以 3,5,7 三数为除数，所以有求解口诀：

三人同行七十稀，

五树梅花廿一枝。

夫妻团圆半月整，

减百零五便得知。

意思是被三除的余数乘以 70，被五除的余数乘以 21，被 7 除的余数乘以 15，三个乘积求和后，若大于 105 则减 105，最后小于 105 的得数即为结果。如三个余数分别为 2,0,4，则有

2×70+0×21+4×15=200,

200−105=95

所以结果是 95。

程序 No. 3－30 是对“韩信点兵”题意的扩展，所以比原题的应用范围更广，更有趣味。

在 No. 3－30 中，如果我们欲查找变量 D，则键入

FIND“D”　↙　则打印

30 D=A * B * C

40 FOR N=1 TO D

若欲查找“MOD”则键入

FIND “MOD”　　↙　则打印

50 IF N MOD A=X THEN IF N MOD B=Y THEN IF N MOD C=Z THEN 70

可见使用查找指令查找某些内容是极为方便的，尤其在较大的程序中将更为优越。

到此为止，F BASIC 语言的基本指令已介绍完毕，加上第二章的七条，共计介绍了 37 条指令。在这 37 条指令中，有可以直接执行的命令，也有必须编在程序中才能执行的函数和语句。运用这些指令已可以编制一些简单的程序。还有一些字符处理、函数及系统控制指令将作为 F BASIC 语言的深入和提高放在第五章中介绍。

第四章　F BASIC的画面控制指令

F BASIC语言有着独特的背景画面设计和卡通画面控制功能。它独特的画面控制指令是任何BASIC语言难以比翼的。使用该部分指令，可以巧妙地设计背景画面(BG GRAPHIC)、灵活地控制卡通图案(SPRITE)的动作。

BG GRAPHIC是英文Background Graphic的缩写，意思为利用背景绘图。SPRITE在英文中是幽魂、小精灵的意思，在此表示卡通图案。为叙述方便，我们把BG GRAPHIC简称为BG，把背景图形面称为BG面；把卡通图案称为SP，把卡通图案面称为SP面。

在画面控制指令中有一部分是F BASIC语言所独有的，其中有些指令目前尚没有任何资料介绍。这些指令对新老程序员来说都较陌生，对初学者来说则更难接受。因此，本文拟以较多的示例，循序渐进，分别介绍该部分指令的功能和使用。

4.1　清屏命令

在程序的编写、输入、调试过程中，为便于阅读和纠错，经常要擦去没有关系的程序，仅留下待编辑的语句，使显示的内容清晰醒目。另外，在程序运行前和运行中，往往也要把屏幕上的内容全部或部分擦去，以突出运算结果。这一擦去原显示内容的过程叫做清屏，清屏使用的命令则称为清屏命令。

4.1.1 清屏命令(CLS)

格式:CLS

类型:语句或命令

简写:CL.

功用:清屏,并使光标定位于屏幕左上角。

清屏之后,用户就可以从屏幕上面开始输入新的信息和画面。

注释:CLS 源于 Clear Screen,是清除屏幕的意思,示例:

```
10 REM "CLS"
20 FOR I=0 TO 500
30 PRINT "#";:NEXT
40 PRINT :PRINT "PAUSE 500 THEN CLS"
50 PAUSE 500:CLS
RUN ↙
```

顺序打印 500 个"#",最后显示 PAUSE 500 THEN CLS,即"延时 500 点,然后清屏。"延时一段时间,清屏,光标移到左上角。

说明:

1. CLS 语句仅限于清屏,并不会清除程序中的变量

2. CLS 一般编写在程序中作为语句使用。作为命令使用的场合不太多。(因即时清屏一般使用清屏操作命令较方便。)

4.1.2 即时清屏命令

所谓即时清屏就是以命令方式直接清屏。直接清屏虽可使用 CLS 命令,但由于要键入三个字母回车后才能清屏,操作烦琐,速度较慢。一般直接清屏都使用即时清屏。即时清屏命令有四个:

1. Shift+Home 全清屏命令。

同时按 Shift 和 Home 键可达到 CLS 相同的效果。

2. Ctrl+L 全清屏命令。

同时按 Ctrl 和 L 键也可达到 CLS 命令的相同效果。

3. Ctrl+Z 半清屏命令

同时按 Ctrl 和 Z 键可以清除光标以后所有画面。它与全清屏命令不同之处是清屏后光标并不回到屏幕左上角而停留在原处。

4. Ctrl+E 行清除命令

同时按 Ctrl 和 E 键可以清除光标所在语句行的全部语句。当光标所在语句行为一行时则只清一行；若为两行时则清两行；若为三行时清三行；……。清除后光标仍留原处。

4.2 光标控制指令

光标控制指令有三条：一条是设定光标位置语句(LOCATE)，第二条是测试光标水平座标指令(POS)，第三是测试光标垂直座标指令(CSRLIN)。

4.2.1 设定光标语句(LOCATE)

格式：LOCATE 〈行，列〉

类型：语句或命令

简写：LOC.

功用：在程序的执行期间对光标进行控制。

LOCATE 语句为用户在程序执行期间对光标进行各种控制提供了有效手段。它可对输出到屏幕上的数据或图像进行定位显示。

注释：LOCATE 为英文定位和放置的意思。

当该语句执行后，光标便被定位到行与列所确定的交点上。行和列用一对数值表示，可以取变量形式，也可以是表达式。行的最大取值范围为 0～27，列的最大取值范围为 0～23，示例：

```
LOCATE 0,10↙
```

光标在座标(0,10)处闪烁。

```
10 REM "LOCATE"
20 FOR,I=0 TO 20
30 LOCATE I,I :PRINT "#"
```

```
40 NEXT
RUN ↙ 打印
#
  #
    #
      #
        #
          #
            #
              #
                #
                  #
                    #
                      #
                        #
OK
□
```

可见，由 LOCATE 语句控制了“#”的打印位置。

例 4.1 编一程序在屏幕的左下角打印一三角形。

```
5 REM "No. 4—1"
10 CLS
20 FOR A=0 TO 23
30 X=0: Y=0
40 X=X+1: Y=Y+A+1
50 IF Y>20 THEN 70
60 LOCATE X,Y:PRINT "*":GOTO 40
70 NEXT
80 LOCATE 0,0
RUN ↙
```

打印如下图形(见图 4—1)。

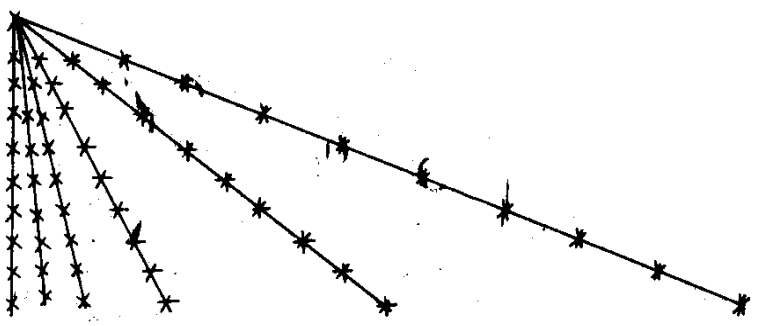


图 4—1

例 4.2 编一程序打印菱形。

```
5 REM "No . 4--2"
10 CLS
20 FOR I=0 TO 10
30 LOCATE 13+I,I: PRINT "$"
40 LOCATE 13-I,I: PRINT "$"
50 LOCATE 13+I,20-I:PRINT "$"
60 LOCATE 13-I,20-I:PRINT "$"
70 NEXT:LOCATE 0,0
```

RUN ↙ 打印一菱形图案。

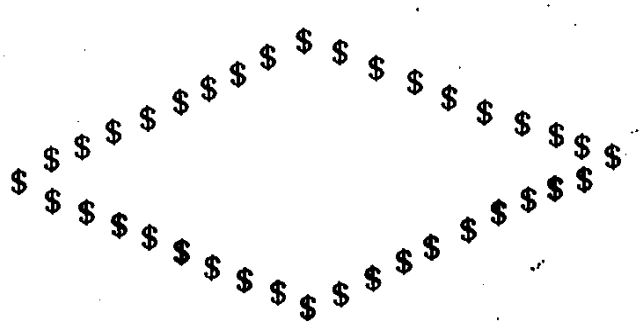


图 4—2

注意——使用 LOCATE 命令设定闪烁光标的位置时，X 的取值恒为 0，即不论 X 取何值，计算机一律把 X 当做 0 处理。这是因为闪烁光标为输入语句的起点，而每行的输入起点必然在 X＝0 的位置，所以 LOCATE 0,15 和 LOCATE 10,15 效果是一样的。如：

```
LOCATE 10,15 ↙
OK
□
LOCATE 20,20 ↙
OK
□
```

光标恒位于 X＝0 位置。

4.2.2 测试光标水平位置(POS)

格式:POS(n)

类型:函数

功用:确定光标所在的水平位置。

注释:POS 取自于 position,为位置的意思。

n 是一个哑变量,无实际意义,可取 32768～32767 之间的任意值。一般取 0。

示例:

```
10 REM "POS"
20 CLS
30 FOR X=0 TO 25
40 LOCATE X,0
50 PRINT POS (0)
60 PAUSE 20
70 NEXT
RUN ↙
```

从左向右依次打印 0、1、2、……25。即光标的水平座标。

4.2.3 测试光标垂直位置(CSRLIN)

格式:CSRLIN

类型:函数

简写:CSR.

功用:确定光标所在的垂直位置。

与 POS 联合使用,可随时测定出光标的行、列位置。

注释:CSRLIN 源于 Cursor(光标)和 Line(行)二词,是光标在哪一行的意思。

CSRLIN 取值范围为 0～23。

示例 1

```
10 REM "CSRLIN"
20 CLS
```

```
30 FOR X=0 TO 10
40 LOCATE X+1,X
50 PRINT POS(0)",";CSRLIN
60 PAUSE 20
70 NEXT
RUN ↙ 打印
 1 , 0
  2 , 1
   3 , 2
    4 , 3
     5 , 4
      6 , 5
       7 , 6
        8 , 7
         9 , 8
          10 , 9
           11 , 10
```

示例 2

```
5 REM "CSRLIN2"
10 CLS:X=0:Y=0
20 X=X+1:Y=Y+5
30 IF X=20 THEN END
40 IF Y>23 THEN Y=0
50 LOCATE X,Y
60 PRINT POS(0)",";CSRLIN
70 PAUSE 20:CLS
80 GOTO 20
RUN ↙
```

闪烁打印光标的各组座标。

4.3 字符—ASCII 码转换指令

字符—ASCII 码转换指令有两条：字符转换成 ASCII 码函数

指令(ASC)和 ACSII 码转换成字符函数指令(CHR $)。ASC 和 CHR $ 为两条相互逆操作指令。

4.3.1 字符转 ASCII 码指令(ASC)

格式:ASC(字符列)

类型:函数

简写:AS.

功用:将字符串或串变量中第一个字符转换成相应的 ASCII 十进制代码。

在计算机上,每个字符都对应一个数字代码,这些代码通常采用"美国信息交换标准代码"的形式。当我们因某种需要想得到某一字符的代码时,就可以使用这一函数。

注释:ASC 取自于 American Standard Code FOR Information Interchange 的缩写,为求 ASCII 码的意思。

字符列是被转换的对象,它可取串变量或串常数。但不管它有多长,ASC 只转换其中的首字符。

示例

```
10 REM "ASC"
20 INPUT "PLEASE INPUT A CHARACTER!";A $
30 A=ASC(A $)
40 PRINT A $ "'s ASCII CODE IS" A
50 PRINT:GOTO 20
RUN ↙ 打印提示
PLEASE INPUT A CHARACTER!? □
提示用户输入字符,若键入 A ↙
A's ASCII CODE IS 65
PLEASE…… 键入 B ↙
B's ASCII CODE IS 66
PLEASE…… 键入 ABC ↙
ABC's ASCII CODE IS 65
```

PLEASE…… 键入 A32B76CDEFG ↙

A32B76CDEFG' s ASCII CODE IS 65

PLEASE……

可见不论键入一个字符或键入多个字符，ASC 函数只给出首字符的 ASCII 码。

4.3.2 ASCII 码转字符指令(CHR $)

格式：CHR $ (X)

类型：函数

简写：CH.

功用：将某个十进制 ASCII 码转换成对应的字符。

ASCII 码与 ASCII 字符是一一对应的。代码是机器内部对数或字符的表示形式。

注释：CHR $ 取自于 Character(字符)的缩写，后面又加了美元号($)，意思是把 ASCII 码换为字符。

X 是一个 ASCII 代码，CHR $ 所生成的是这个代码所对应的字符。X 取值范围为 0～255 的整数，它可以为数值常数，也可以为数值变量或表达式。

示例

```
10 REM "CHR $ "
20 INPUT "PLEASE INPUT AN ASCII CODE";A
30 A $ =CHR $ (A)
40 PRINT"'SCHR. IS"A $
50 PRINT:GOTO 20
RUN ↙
PLEASE INPUT AN ASCII CODE?
```

提示请输入 ASCII 代码。键入 65 ↙

```
65's CHR. IS A
PLEASE…… 键入 90 ↙
90's CHR. IS Z
```

PLEASE……

例 4.3 编程序打印 F BASIC 系统的 ASCII 码和对应的字符。

```
10 REM "No. 4—3"
20 FOR I=0 TO 255
30 A$=CHR$(I)
40 PRINT I"="A$,
50 PRINT:NEXT
RUN ↙ 打印
0=
1=
……
33= !
34= "
……
65= A
66= B
……
```

注——在随机手册中一般仅刊印背景图形库中的 104 种图案(见封二),这在 BG GRAPHIC 绘图时调用极为方便,但在编程中欲在屏幕上的某个位置打印某个图形时则很难如意。通过例 4.3 的运行结果,我们找出了背景图案对应的 ASCII 代码,这给我们随意调用背景图案带来了方便。如欲在屏幕的(10,10)处向下画五节梯子,我们可编如下程序实现。

```
10 CLS
20 FOR I=1 TO 5
30 LOCATE 10,10+I
40 PRINT CHR$(210)
50 NEXT
RUN ↙ 可打印出如下图形(见图 4—3)
```

图 4—3

为便于读者使用，特把 104 种背景图案对应的 ASCII 码列出。（仍采用背景图形库图表的格式。）

表 4—1　背景图形对应的 ASCII 码一览表

	1	2	3	4	5	6	7	8
A	0	1	2	3	4	5	6	7
B	8	9	10	11	12	13	14	15
C	16	17	18	19	20	21	22	23
D	24	25	26	27	28	29	30	31
E	184	185	186	187	188	189	190	191
F	192	193	194	195	196	197	198	199
G	200	201	202	203	204	205	206	207
H	208	209	210	211	212	213	214	215
I	216	217	218	219	220	221	222	223
J	224	225	226	227	228	229	230	231
K	232	233	234	235	236	237	238	239
L	240	241	242	243	244	245	246	247
M	248	249	250	251	252	253	254	255

说明：在表 4—1 中，第 E 行至第 M 行的 72 个图形可以用 CHR $ 函数调出，通过 LOCATE 语句打印在屏幕的任意位置上。但第 A 行至第 D 行的 32 个图形，虽然我们在表 4—1 中分别标上了 0～31，共 32 个 ASCII 码，但用 CHR $ 函数却不能在屏幕上打印。关于这 32 个图形的调用将在以后介绍的 CGEN、SPRITE 语句的示例中介绍。

示例

```
10 CLS
20 LOCATE 6,0
30 PRINT CHR$(216);CHR$(218)
40 LOCATE 7,1
50 PRINT CHR$(216);CHR$(217);CHR$(218)
60 END
```

RUN ↙ 在屏幕的上方打印出一朵白云。

例 4.4 试编写一个大、小写字母转换程序。

```
10 REM "No. 4—4"
20 CLS
30 INPUT "PLEASE INPUT A LETTER";C$
40 A=ASC(C$)
50 IF A>97 AND A<=122 THEN A=A-32: GOTO 70
60 IF A>=65 AND A<=90 THEN A=A+32
70 L$=CHR$(A)
80 PRINT C$"CHANGES TO"L$
90 PRINT:GOTO 30
```

RUN ↙

PLEASE INPUT A LETTER ? □

提示请输入一个字母。键入 A ↙

A CHANGES TO a 大写的 A 变为小写的 a。

PLEASE…… 键入 b ↙

b CHANGES TO b

PLEASE…… 键入 M ↙

M CHANGES TO m

PLEASE……

为便于读者熟悉、掌握、使用 F-BASIC 语言的符号系统，特列出《符号代码表 B》(见表 4—2)。(表中有关图形采用表 4—1 中的座标，如背景图形库中第一行第一个图形用"A1"表示，第二个用"A2"表示，……最后一个用"M8"表示。)

表 4—2 《符号代码表 B》(用于 BG 面)

代码 (10 进)	代码 (16 进)	对应字符	代码 (10 进)	代码 (16 进)	对应字符	代码 (10 进)	代码 (16 进)	对应字符
0	00	A1	36	24	$	72	48	H
1	01	A2	37	25	%	73	49	I
2	02	A3	38	26	&	74	4A	J
3	03	A4	39	27	,	75	4B	K
4	04	A5	40	28	(	76	4C	L
5	05	A6	41	29	)	77	4D	M
6	06	A7	42	2A	*	78	4E	N
7	07	A8	43	2B	+	79	4F	O
8	08	B1	44	2C	,	80	50	P
9	09	B2	45	2D	-	81	51	Q
10	0A	B3	46	2E	.	82	52	R
11	0B	B4	47	2F	/	83	53	S
12	0C	B5	48	30	0	84	54	T
13	0D	B6	49	31	1	85	55	U
14	0E	B7	50	32	2	86	56	V
15	0F	B8	51	33	3	87	57	W
16	10	C1	52	34	4	88	58	X
17	11	C2	53	35	5	89	59	Y
18	12	C3	54	36	6	90	5A	Z
19	13	C4	55	37	7	91	5B	[
20	14	C5	56	38	8	92	5C	¥
21	15	C6	57	39	9	93	5D	]
22	16	C7	58	3A	:	94	5E	∧
23	17	C8	59	3B	;	95	5F	—
24	18	D1	60	3C	<	96	60	、
25	19	D2	61	3D	=	97	61	a
26	1A	D3	62	3E	>	98	62	b
27	1B	D4	63	3F	?	99	63	c
28	1C	D5	64	40	@	100	64	d
29	1D	D6	65	41	A	101	65	e
30	1E	D7	66	42	B	102	66	f
31	1F	D8	67	43	C	103	67	g
32	20	空格	68	44	D	104	68	h
33	21	!	69	45	E	105	69	i
34	22	"	70	46	F	106	6A	j
35	23	#	71	47	G	107	6B	k

续表 4—2

代码（10 进）	代码（16 进）	对应字符	代码（10 进）	代码（16 进）	对应字符	代码（10 进）	代码（16 进）	对应字符
108	6C	l	135	87	八	162	A2	ヂ
109	6D	m	136	88	九	163	A3	ツ
110	6E	n	137	89	十	164	A4	デ
111	6F	o	138	8A		165	A5	ド
112	70	p	139	8B		166	A6	バ
113	71	q	140	8C		167	A7	ビ
114	72	r	141	8D		168	A8	ブ
115	73	s	142	8E		169	A9	ベ
116	74	t	143	8F		170	AA	ボ
117	75	u	144	90		171	AB	パ
118	76	v	145	91		172	AC	ポ
119	77	w	146	92		173	AD	プ
120	78	x	147	93		174	AE	ペ
121	79	y	148	94		175	AF	ポ
122	7A	z	149	95		176	B0	□
123	7B	{	150	96		177	B1	
124	7C	\	151	97	ガ	178	B2	
125	7D	}	152	98	ギ	179	B3	
126	7E	⌴	153	99	グ	180	B4	C
127	7F	⌒	154	9A	ゲ	181	B5	×
128	80	一	155	9B	ゴ	182	B6	÷
129	81	二	156	9C	ザ	183	B7	）
130	82	三	157	9D	ジ	184	B8	E1
131	83	四	158	9E	ズ	≀	≀	≀
132	84	五	159	9F	ゼ			
133	85	六	160	A0	ゾ			
134	86	七	161	A1	タ	255	FF	M8

表 4—2 中代码 184～255 为背景图形库中第五行至第十三行共七十二个背景图案，已在表 4—1 中列出，故此处从略。具体的对应关系见附录。

4.4 画面控制指令

所谓画面控制是指屏幕的显示方式设置、BG 面及 SP 面上的

符号分配和对画面显示内容的测试。画面控制指令有三条，分别为SCREEN、CGEN 和 SCR $ 。

4.4.1 设置显示方式语句(SCREEN)

格式：SCREEN 〈M，N〉

类型：语句或命令

简写：SC.

功用：设置屏幕的显示方式。

通过 SCREEN 语句可将屏幕设置成高分辨图形方式或高、低分辨及文本混显方式。

注释：SCREEN 为屏幕的意思。

1. 由 M、N 决定屏幕的显示方式。M、N 可以是数字，变量或表达式，取值均为 0、1。当 M＝N 时，屏幕为文本、低分辨、高分辨混合显示方式。当 M≠N 时，屏幕为高分辨显示方式。

2. 语句中 M＝0 时，执行 SCREEN 不清屏；当 M＝1 时，执行 SCREEN 则清屏。如执行

SCREEN 1，1 ↙

则清屏，置混显方式。

3. 另外，当 M＝N 时，执行 SCREEN 后，显示语句执行完毕提示符“OK”和光标；当 M≠N 时，执行 SCREEN 后则不显示“OK”和光标。这一功能在设计游戏画面时极为有用，它可避免出现“OK”和光标而破坏画面。如

SCREEN 0，0 ↙

不清屏，显示“OK”和光标，且光标回到左上角。

SCREEN 0，1 ↙

则不清屏，也不显示“OK”和光标(注意这时形同死机，必须用热启动才能解除)。

SCREEN 1，0 ↙

清屏，不出现“OK”和光标，屏幕上一片空白，干干净净。

在以后的叙述中，我们还会看到 SCREEN 的重要作用，它和 VIEW 语句配合，可以实现“前台显示、屏后绘图”，从而在游戏程序的背景换场中发挥巨大作用。

4.4.2 画面符号配置语句(CGEN)

CGEN 是 F BASIC 语言所特有的语句

格式：CGEN〈n〉

类型：语句或命令

简写：CGE.

功用：决定 BG 面、SP 面上符号的分配。

CGEN 规定了哪些内容显示于背景面，哪些内容显示于 SP 面。

注释：CGEN 为 CHARACTER GENERATION 的缩写，CHARACTER 为分配种类字符、符号，GENERATION 表示产生、发生的意思。

n 是符号分配的组合范围，它可以是数值常数、数值变量、数值表达式，其取值范围为 0～3。

说明：CGEN 语句的使用规定如下表

表 4—3 CGEN 指令中 n 的分配

n	0	1	2	3
底背景状态	突起	突起	黑色	黑色
背景画面	变为阴体的字母	变为阴体的字母	正常	正常
字母状态	变为图案	变为图案	正常	正常
卡通状态	正常显示	变为背景图案	正常显示	变为背景图案
光标状态	变为阴A字母	变为阴A字母	正常显示	正常显示

表 4—3 说明了 n＝0、n＝1 的情况使用较少，n＝2 的情况使用最多，所以当用户不设定 n 时(即不使用 CGEN 语句)系统自动设定为 CGEN2。

示例：

```
10 REM "CGEN"
20 CLS
30 FOR I=32 TO 255
40 PRINT CHR$(I);
50 NEXT
60 PAUSE 100:CGEN 0
70 PAUSE 100:CGEN 1
80 PAUSE 100:CGEN 2
90 PAUSE 100:CGEN 3
RUN ↙
```

首先打印字符一览表。延时一段时间，变为突起底背景，同时所有的背景图案均变为阴体字母，而所有字母则变为图案。再延时一段时间，则一切变为正常。(由于没有 SP，所以不能区分 CGEN0、CGEN1 的差别，也不能区分 CGEN2、CGEN3 的差别。待介绍卡通显示指令时再举例演示。)

4.4.3 测试 BG 面函数(SCR$)

格式：SCR$(X,Y,S)

类型：函数

简写：SC.（在 BS 状态中使用）

功能：求出 BG 画面中所显示的文字或图画的数值。

当需要了解画面上某一位置(X,Y)处所显示的内容或其位置的配色号码时，使用 SCR$ 函数极为有效。

注释：SCR$ 源于 SCREEN，即屏幕，$ 表示字符或字串，意思是测试屏幕字符。

X 为测试点水平方向的座标，取值范围为 0～27。它可以是常

数、变量或表达式。

Y 为测试点垂直方向的座标，取值范围为 0～23。它可以是常数、变量或表达式。

S 为求测试点的配色号码软开关。S 等于 1 表示开关打开，求配色号码；S 等于 0 时表示关闭开关，不求配色号码。一般 S＝0 时可以省略 S。

示例

```
10 REM "SCR$"
20 CLS
30 LOCATE 0,10
40 PRINT "FAMILY COMPUTER"
50 PRINT ". . . . . ."
60 LOCATE 10,15
70 PRINT SCR$(0,10)
80 A$=SCR$(1,10)
90 PRINT A$
100 C$=SCR$(1,10,1)
110 PRINT "COLOR="ASC(C$)
RUN ↙ 打印
FAMILY COMPUTER
. . . . . .
PA
COLOR=0
OK
```

4.5 配色控制指令

F BACIC 系统共有 52 种颜色代码供用户任意组合。使用染色指令(COLOR)、置色指令(PALET)和选色指令 CGSET，可以任意组合 BG 面、SP 面的配色，能够同时改变背景底色、背景图案、卡通

图案三者的色彩、色调，从而使画面绚丽多彩、变幻莫测。

4.5.1 染色指令(COLOR)

格式：COLOR〈X，Y，n〉

类型：语句

简写：COL.

功用：对屏幕上某个位置的背景图案染色。

注释：COLOR 是颜色的意思

在 COLOR 语句的作用下，可对屏幕任一位置的背景图案染色。染色位置由 x、y 决定。x 为屏幕水平方向的位置座标，取值范围为 0～27；Y 为垂直方向的位置座标，取值范围为 0～23(见图 4—4)；n 为选定的配色号码，取值范围为 0～3。x、y、n 均可为常数、变量或表达式。

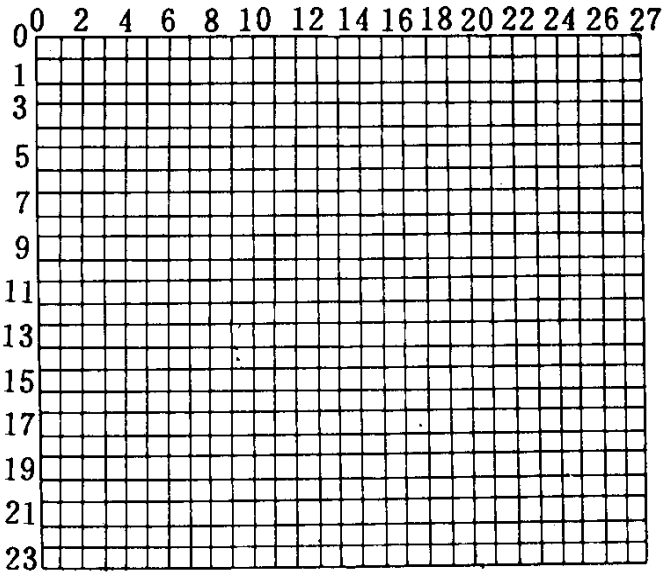


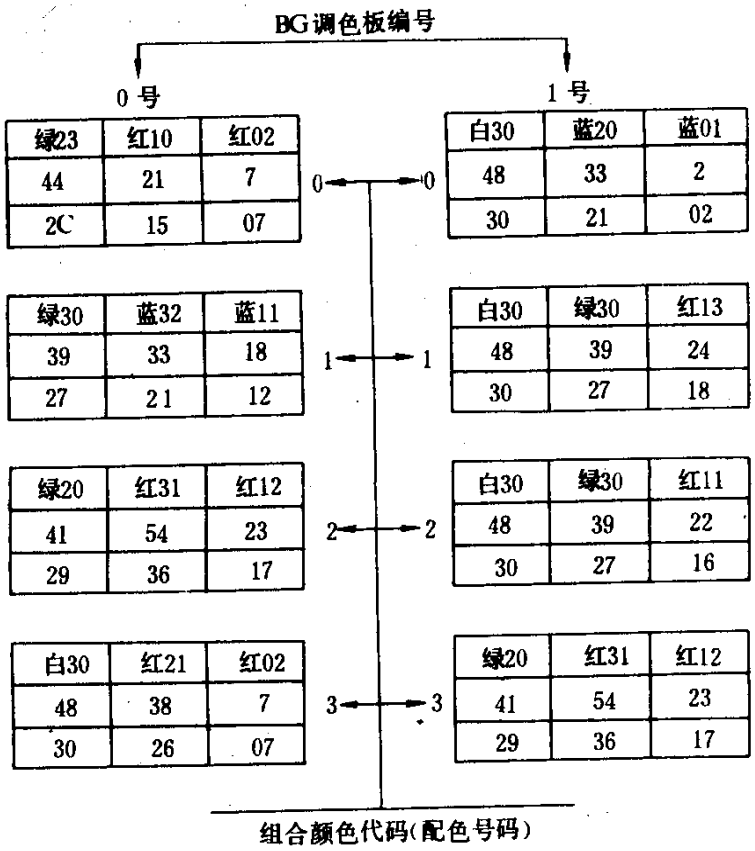
图 4—4 COLOR 指令作用的屏幕分成 16×16 点方块的示意图

配色号码 n 为由选色指令 CGSET 所选定背景调色板中的配色编号。背景用色有两块背景调色板，每块调色板中有四种组合颜色，每种组合可选定三种色彩。调色板编号为 0、1；组合颜色编号为 0～3。(详见背景配色号码表 4—4 和附录)，一般称为配色号

码。

注释:图 4—4 表示可把屏幕按 4 格一块分成一个个小方块,使用 COLOR 指令指定 x,y 座标,可把小方块的内部按配色号码加以染色。染色用的配色号码见表 4—4。

表 4—4 背景配色号码表



若以色板编号在前、颜色代码在后,则上表中八种颜色组合可用八个数字代表,分别为:00、01、02、03、10、11、12、13。

表 4—4 中每块颜色组合由三种颜色代码组成。其中第一行为颜色、第二行为十进制颜色代码、第三行为十六进制颜色代码。在第一行的颜色后的数字表示该种色彩的浓淡和明暗,颜色代码的分配见表 4—5。

表 4—5　52 种颜色代码

		0 列		1 列		2 列		3 列	
行别	系别	十六进制	十进制	十六进制	十进制	十六进制	十进制	十六进制	十进制
0	灰、白系	00	0	10	16	20	32	30	48
0	兰系	01	1	11	17	21	33	31	49
1		02	2	12	18	22	34	32	50
2		03	3	13	19	23	35	33	51
3		04	4	14	20	24	36	34	52
0	赤系	05	5	15	21	25	37	35	53
1		06	6	16	22	26	38	36	54
2		07	7	17	23	27	39	37	55
3		08	8	18	24	28	40	38	56
0	绿系	09	9	19	25	29	41	39	57
1		0A	10	1A	26	2A	42	3A	58
2		0B	11	1B	27	2B	43	3B	59
3		0C	12	1C	28	2C	44	3C	60
0	黑系	0D	13	1D	29	2D	45		
1		0E	14	1E	30	2E	46-		
2		0F	15	1F	31	2F	47		

暗←——————————→明

从表 4—5 中可见兰系、红系、绿系每系有 16 种颜色变化，灰白系有 4 种，黑系有 9 种，共 61 个代码。每个代码由行、列座标确定位置。每颜色系的行号从 0～3，行号越大颜色越深；列号从 0～3，列号越大，色调越明亮。如："红 00"为暗色浅红、"红 33"为亮色深红。

背景和卡通图案的颜色均可从表 4—5 中选取。

示例：

```
10 REM "COLOR"
20 CLS
30 FOR I=0 TO 447
40 PRINT CHR$(195);
50 NEXT
```

```
60 FOR C=0 TO 3
70 COLOR 5+C*3,5+C*2,C
80 NEXT
90 LOCATE 0,20
100 END
RUN ↙
```

分别在(5,5)、(8,7)、(11,9)、(14,11)四点染出四个色块。

例 4.7 编程演示色彩的变化情况

```
5 REM "No. 4--7"
10 FOR I=0 TO 671
20 PRINT CHR$(255);:NEXT
30 X=RND(28): Y=RND(24): Z=RND(1)
40 COLOR X,Y,Z
50 GOTO 30
RUN ↙
```

屏幕上先打印一兰色底面背景，然后产生各种色彩块的千变万化，形成五彩斑斓的屏幕万花筒。

说明：通过示例和例 4.7 的运行结果可以得出 COLOR 语句的染色规律和运行要求。

1. COLOR 语句只能对背景图案染色而不能对底背景染色。

2. COLOR 语句执行后，不仅对指定的位置染色，而且还把指定位置的上、右、上右三个位置染色。如 COLOR 10,10,3 执行后(见下图)，不仅染了(10,10)块，而且把(9,10)、(9,11)、(10,11)三块即 A、B、C、D 四块全部染上配色号码 3 的颜色。

	第 10 列	第 11 列
第 9 列	A	B
第 10 列	C	D

3. 由于 COLOR 语句一次就染四个块(16×16 点)，所以它染色的色块较大。

例 4.8　用不同的颜色作为金砖，垒起一座金山。要求金山共七层，最下层七块、最上层一块，共计 28 块金砖。

```
10 REM "No. 4—8"
20 CLS: A$=CHR$(255):A$=A$+A$
30 FOR J=1 TO 7
40 X=(7-J)*2
50 FOR I=1 TO J
60 LOCATE X, J*2+3:PRINT A$
70 LOCATE X, J*2+4:PRINT A$
80 COLOR X, J*2+4,RND(4)
90 X=X+4:NEXT
100 NEXT
110 LOCATE 0,22
RUN ↙ 打印金山图形(见图 4—5)
```

可见使用 COLOR 语句染色的方法画出的金山，金砖块较大，且色彩变化仅有四种。为使画出的金山更形象，金砖的色彩、形态更多变，我们可选取多种背景图案画金砖。直接使用背景图案的原色彩，从而省去染色操作，见例 4.9。

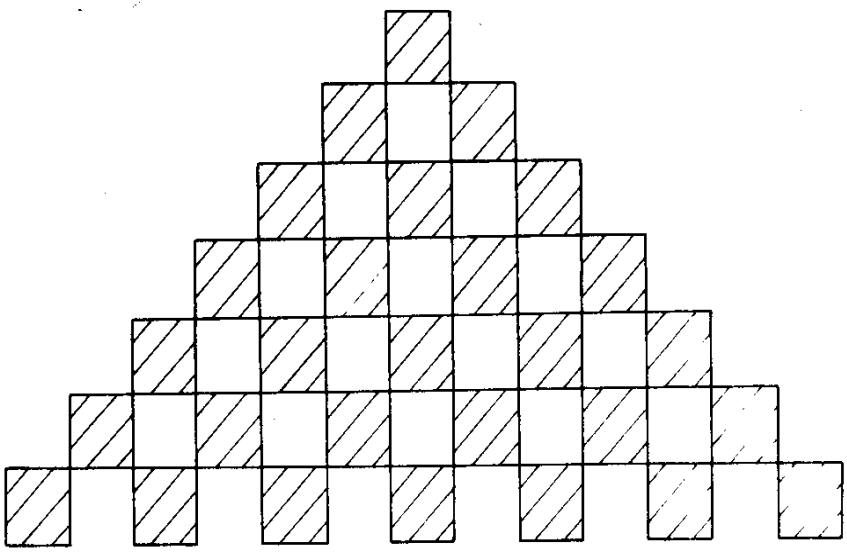


图 4—5 用 COLOR 画金山

例 4.9

```
10 REM "No. 4—9"
20 CLS:DIM A(13)
30 FOR I=0 TO 13:READ A(I):NEXT
40 DATA 244,255,255,252,252,252,250,250,250,246,246,246,246,
   246
50 FOR J=1 TO 7
60 X=(7-J)*2
70 FOR I=1 TO J
80 A$=CHR$(A(RND(14))):A$=A$+A$
90 LOCATE X,5+J:PRINT A$
100 X=X+4:NEXT
110 NEXT
120 LOCATE 0 22: END
```

RUN ↙ 画出的金山见图 4—6。

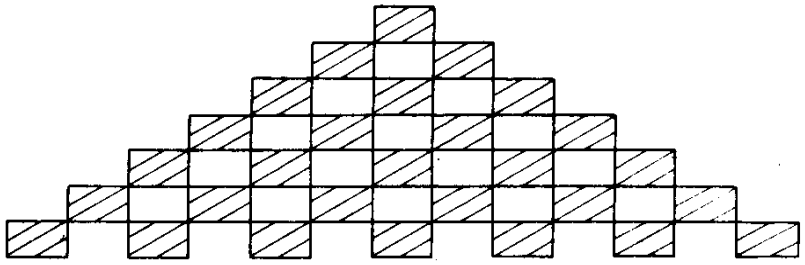


图 4—6 用不同的图形画金山

4.5.2 背景置色语句(PALETB)

格式:PALETB n,C1,C2,C3,C4

类型:语句

简写:PAL. B

功用:将 BG 调色板中配色号码所代表的颜色组合重新设定为任意的颜色代码。

使用 PALETB 语句,可将背景任一调色板中的四个配色号码的内容,重新设定为新的颜色组合。(调色板号由 CGSET 语句选定。)

注释：PALETB 取自于 PALETTE BACKGROUND，为对背景配色的意思。

n 为配色号码，取值范围 0～3，它可以是数值、变量或表达式。（配色号码就是表 4—4《背景配色号码表》中的颜色组合代码。）

C1 是指定底背景颜色的颜色代码。开机后，底背景的颜色由系统自动设定为黑色。当我们欲改变底背景的颜色时，可以通过 C1 的取值来确定。C1 为表 4—5《52 种颜色代码》表中的任一种颜色代码。但在指定底背景颜色时，必须先打开底背景置色软开关。该开关通过改变配色号码 n 的取值来控制。n＝0 为打开开关；n≠0 为关闭开关。C1 的取值范围为 0～60。它可以是数值常数，可以是变量，也可以是表达式。

C2、C3、C4 分别指定《52 种颜色代码表》中的任三种颜色代码。它们分别对应配色号码板中每组颜色组合的左、中、右三种颜色代码。它们的对应关系如下：

↑	↑	↑
C2	C3	C4

表中的箭头分别表示 C2、C3、C4 所对应的颜色。C2、C3、C4 的取值范围均为 0～60，它们可以是数值，可以是变量，也可以是表达式。

C1～C4 可以用十进制数表示；也可以用十六进制数表示。用十六进制数表示时要注意在数字前边添“&H”。

示例：

```
PALETB 0,7,44,21,7
```

运行后，把底背景置暗红色。

```
PALETB 3,&H0F,&H30,&H26,&H12
```

运行后，把背景调色板中 3 号配色号码中的 C2、C3、C4 颜色代码分别换为 48、38、18。

4.5.3 卡通置色语句(PALETS)

同背景图案一样，卡通图案也有它自己专用的调色板。

卡通图案的调色板共有三块，编号分别为 0、1、2。每个调色板也有四个配色号码，分别为 0、1、2、3。每个配色号码也同样由三个颜色代码组成，它也同样使用表 4—5 中的 52 种颜色代码。

系统中自动设定的配色号码内容详见表 4—6。

如果系统中原定配色号码的颜色组合不符合要求，我们可以重新设定。重新设置 SP 配色号码的语句是 PALETS。

表 4—6　卡通配色号码表

SP调色板编号

配色号码	0号			1号			2号			配色号码
0	红31	红11	蓝01	白30	红11	蓝00	广130	红21	红13	0
	54	22	2	18	22	1	18	38	18	
	36	16	02	30	16	01	30	26	12	
1	红32	广130	绿20	白10	白00	蓝00	广130	红10	蓝11	1
	55	18	41	16	0	1	18	21	18	
	37	30	29	10	00	01	30	15	12	
2	红30	红20	红12	广130	绿20	绿00	广130	蓝11	红11	2
	53	37	23	18	41	9	18	18	22	
	35	25	17	30	29	09	30	12	16	
3	白30	红22	红11	广130	红11	红02	广130	红21	红10	3
	18	39	22	18	22	7	18	38	25	
	30	27	16	30	16	07	30	26	19	

格式：PALETS n，C1，C2，C3，C4

类型：语句

简写：PAL. S

功用：将 SP 调色板中配色号码所代表的颜色组合重新设定为任意的颜色代码。

使用 PALETS 语句，可将 SP 任一调色板中的四个配色号码的内容，重新设定为新的颜色组合。（调色板号由 CGSET 语句选定。）

注释：PALETS 取自于 PALETTE SPRITE，是对 SP 配色的意思。

n 为配色号码，取值范围 0～3，它可以是数值，变量或表达式。

C1 仍能指定底背景的颜色代码。使用方法见 PALETB。

C2、C3、C4 的作用及使用方法，见 PALETB。

示例：

```
10 REM "PALETS"
20 CGSET 0,0:CLS
30 SPRITE ON
40 DEF MOVE(0)=SPRITE(0,3,1,120,0,0)
50 MOVE 0
60 IF MOVE(0)=-1 THEN 60
70 PALETS 0,47,54,24,25:MOVE 0
RUN ↙
```

玛丽先以原配色(54,22,2)的颜色组合运动(主色为兰、红)，随后变为(54,24,25)的颜色组合运动(主色为绿、红)。

4.5.4 选色指令(CGSET)

格式：CGSET〈m,n〉

类型：语句或命令

简写：CG.

功用：规定 BG 或 SP 面的颜色组合。

注释：m 表示背景用的调色板编号。取值为 0 和 1。m 可以是数值，可以是变量，也可以是表达式。

n 表示 SP 用的调色板编号。取值为 0、1、2。n 也可以取数值、变量、表达式。

说明：

1. 一般在程序的开始语句中首先使用 CGSET 语句，以决定 BG、SP 面的颜色组合。

2. 当 BG、SP 的调色板中颜色组合不符合要求时，要先使用

CGSET 语句选定调色板，再用 PALETS 语句设定配色号码内容。

3. 在用户不设定调色板编号时(即不使用 CGSET 语句)系统自动设定为 CGSET1，1。

4. 在 COLOR 语句中决定的配色号码 n，为由 CGSET 语句选定的调色板中的编号。如

```
CGSET 0,1
COLOR X,Y 1
```

则选定的颜色组合为 39、33、18。

若 COLOR 语句不变，调整 CGSET 语句为 CGSET 1,1

则选定的颜色组合为 48、39、24。

示例

我们把程序 No. 4—8 改写如下：

```
10 REM "No. 4—10"
20 CLS:A$=CHR$(255):A$=A$+A$
30 FOR J=1 TO 7
40 X=(7-J)*2
50 FOR I=1 TO J
60 LOCATE X,J*2+3:PRINT A$
70 LOCATE X,J*2+4:PRINT A$
80 COLOR X,J*2+4,RND(4)
90 X=X+4:NEXT:NEXT
100 PAUSE 100:A=A+1:IF A>1 THEN A=0
110 CGSET A
120 GOTO 30
RUN ↙
```

对比 No. 4—8，No. 4—10 的色彩有了更丰富的变化。

关于四条配色控制指令的具体使用将在介绍卡通运动控制指令中详细介绍。

4.6 卡通显示指令

F BASIC 系统中固化有 16 种卡通图案、58 种姿式。十六种图案的分类如表 4—7。58 种姿式的代码见表 4--8。

表 4—7 卡通图案分类表

编 号	卡通名称	编 号	卡通名称
0	玛莉哥哥	8	杀手卫星
1	莉 莎	9	太空船
2	苍 蝇	10	爆 炸
3	飞 鱼	11	妖 怪
4	企 鹅	12	镭 射
5	火 球	13	乌 龟
6	车	14	螃 蟹
7	太空堡垒	15	飞 鸟

表 4—8 符号代码表 A(主要用于 SP)

十进制代码	十六进制代码	说 明	十进制代码	十六进制代码	说 明
0	00	玛莉哥哥(走 1)	24	18	玛莉哥哥(跌下)
1	01		25	19	
2	02		26	1A	
3	03		27	1B	
4	04	玛莉哥哥(走 2)	28	1C	丽莎(走 1)
5	05		29	1D	
6	06		30	1E	
7	07		31	1F	
8	08	玛莉哥哥(走 3)	32	20	丽莎(走 2)
9	09		33	21	
10	0A		34	22	
11	0B		35	23	
12	0C	玛莉哥哥(跳)	36	24	丽莎(走 3)
13	0D		37	25	
14	0E		38	26	
15	0F		39	27	
16	10	玛莉哥哥(滑)	40	28	丽莎(跳)
17	11		41	29	
18	12		42	2A	
19	13		43	2B	
20	14	玛莉哥哥(背面)	44	2C	丽莎(滑)
21	15		45	2D	
22	16		46	2E	
23	17		47	2F	

续表 4—8

十进制代码	十六进制代码	说明	十进制代码	十六进制代码	说明
48 49 50 51	30 31 32 33	丽莎 (背面)	88 89 90 91	58 59 5A 5B	妖怪 (1)
52 53 54 55	34 35 36 37	丽莎 (跌下)	92 93 94 95	5C 5D 5E 5F	妖怪 (2)
56 57 58 59	38 39 3A 3B	苍蝇 (1)	96 97 98 99	60 61 62 63	企鹅 (步 1)
60 61 62 63	3C 3D 3E 3F	苍蝇 (2)	100 101 102 103	64 65 66 67	企鹅 (步 2)
64 65 66 67	40 41 42 43	飞鱼 (1)	104 105 106 107	68 69 6A 6B	企鹅 (正面)
68 69 70 71	44 45 46 47	飞鱼 (左 2)	108 109 110 111	6C 6D 6E 6F	企鹅 (背面)
72 73 74 75	48 49 4A 4B	飞鱼 (左上 1)	112 113 114 115	70 71 72 73	火球 (1)
76 77 78 79	4C 4D 4E 4F	飞鱼 (左上 2)	116 117 118 119	74 75 76 77	火球 (2)
80 81 82 83	50 51 52 53	飞鱼 (前 1)	120 121 122 123	78 79 7A 7B	车 (左上 1)
84 85 86 87	54 55 56 57	飞鱼 (前 2)	124 125 126 127	7C 7D 7E 7F	车 (左 2)

续表 4—8

十进制代码	十六进制代码	说明	十进制代码	十六进制代码	说明
128	80	车（左上 1）	160	A0	杀手卫星（上）
129	81		161	A1	
130	82		162	A2	
131	83		163	A3	
132	84	车（左上 2）	164	A4	太空船（左）
133	85		165	A5	
134	86		166	A6	
135	87		167	A7	
136	88	车（上 1）	168	A8	太空船（左上）
137	89		169	A9	
138	8A		170	AA	
139	8B		171	AB	
140	8C	车（上 2）	172	AC	太空船（上）
141	8D		173	AD	
142	8E		174	AE	
143	8F		175	AF	
144	90	太空站（1）	176	B0	爆炸（开始）
145	91		177	B1	
146	92		178	B2	
147	93		179	B3	
148	94	太空站（2）	180	B4	爆炸（散发）
149	95		181	B5	
150	96		182	B6	
151	97		183	B7	
152	98	杀手卫星（左）	184	B8	乌龟走路（1）
153	99		185	B9	
154	9A		186	BA	
155	9B		187	BB	
156	9C	杀手卫星（左上）	188	BC	乌龟走路（2）
157	9D		189	BD	
158	9E		190	BE	
159	9F		191	BF	

续表 4—8

十进制代码	十六进制代码	说明	十进制代码	十六进制代码	说明
192	C0	螃蟹(1)	217	D9	音乐板用三音程(1)(2)(3)
193	C1		218	DA	
194	C2		219	DB	
195	C3		220	DC	第一指标
196	C4	螃蟹(2)	221	DD	第二指标
197	C5		222	DE	指示器(1)
198	C6		223	DF	指示器(2)
199	C7		224	E0	向上羽毛笔
200	C8	飞鸟(1)	≀	≀	
201	C9		231	E7	
202	CA		232	E8	向下羽毛笔
203	CB		≀		
204	CC	飞鸟(2)	239	EF	
205	CD		240	F0	1
206	CE		241	F1	2
207	CF		242	F2	3
208	D0	镭射弹(1)(2)	243	F3	4
209	D1		244	F4	COMPUTER的文字集区
210	D2	镭射束(1)(2)	≀	≀	
211	D3		249	F9	
212	D4	镭射刀(1)(2)	250	FA	操作者输入的文字记忆区
213	D5		≀	≀	
214	D6	SP用配色号码(1)(2)(3)	255	FF	
215	D7				
216	D8				

在表 4—8 中，十进制代码 0～207 为除镭射外的十五种卡通的 52 种姿式，每一个姿式对应 4 个代码。在介绍 DEF SPRITE 语句时我们将会看到，使用对应的 4 个代码可令 SP 显示在屏幕的任一位置。代码 208～213 为镭射的 6 种姿式，它们只要一个代码就可令其显示。关于 214 以后代码的含义和作用已超出本章内容范畴，留待以后讨论。

控制卡通显示的指令有五条：开显示指令(SPRITE ON)、选显示指令(DEF SPRITE)、显示定位指令(SPRITE)、消显示指令

(SORITE n)和关显示指令(SPRITE OFF)。使用这五条指令可任意选定卡通类型,灵活地控制其显示的位置,随心所欲地令卡通图案显示或消失。这五条指令都是 F BASIC 所独有的指令。

4.6.1 开显示指令(SPRITE ON)

格式:SPRITE ON

类型:语句或命令

简写:SP. O

功用:打开 SP 显示开关,使 SP 面变成重复显示于背景面的状态。

注释:SPRITE ON 是打开 SPRITE 显示开关的意思。

SPRITE ON 为 SP 在屏幕上显示提供了可能条件,至于 SP 是否显示、显示哪种姿式、在何处显示还要取决于其它语句。

在执行 DEF SPRITE 或 DEF MOVE 语句之前,必须首先执行 SPRITE ON。否则,SP 不能显示于画面之上。

SPRITE ON 语句执行后直到执行 SPRITE OFF 前皆为有效。

示例:

```
SPRITE ON ↙
OK
□
10 SPRITE ON
RUN ↙
OK
□
```

4.6.2 选显示指令(DEF SPRITE)

格式:DEF SPRITE n,(A,B,C,D,E)=〈字符表达式〉

类型:语句

简写:DE. SP.

功用:选定显示于 SP 面的 SP。

注释:

1. DEF SPRITE 源于 Define a sprite，是定义一个卡通图案的意思。

2. n 是显示 SP 的编号，由用户随意设定。n 的取值范围为 0～7，所以，在同一画面最多可同时显示 8 个 SP。

3. A 为配色号码，取值范围 0～3。它是由 CGSET 语句选定的 SP 调色板中的任一组颜色组合代码。A 可以取常数、变量或表达式。

4. B 为符号合成类型，取值范围为 0～1。它标志着显示于 SP 面的 SP 的大小。B＝0，则显示的 SP 为 8×8 点阵(即一个字符的大小)。B＝1，则 SP 为 16×16 点阵(四个字符的大小)。

5. C 表示显示的优先级。C＝0 表示 SP 显示在背景图案的前面；C＝1 则 SP 在背景图案的后面显示。

6. D 表示显示的卡通在 X 轴方向的反转指示。取值 0～1。D＝0 表示显示 SP 的方向与 SP 符号表(见附录)中的方向一致；D＝1 则显示的 SP 左右逆向。(见图 4—7a、b)。

7. E 表示显示的卡通在 Y 轴方向的反转指示。取值范围 0～1。E＝0 表示与符号表相同；E＝1 则上下逆向。(见图 4—7a)。

8. 字符表达式：

字符表达式有两种形式：一种使用 CHR $ 语句组成，如 CHR $ (0)＋CHR $ (1)＋CHR $ (2)＋CHR $ (3)(选定玛莉走 1)；另一种形式用字符表示，如“DEFG”(选定飞鱼左 2)。可以使用字符表示的理由是：飞鱼对应的字符表达式实际上应为 CHR $ (68)＋CHR $ (69)＋CHR $ (70)＋CHR $ (71)，而 CHF $ (68) 正好对应字母“D”，即 D＝CHR $ (68)。同理，有 E＝CHR $ (69)、F＝CHR $ (70)、G＝CHR $ (71)。所以 CHR $ (68)＋CHR $ (69)＋CHR $ (70)＋CHR $ (71)＝D＋E＋F＋G＝“DEFG”。

9. B、C、D、E 都同 A 一样可以是数值、变量和表达式。

反转符号定义示意图(为便于绘制，仅以字母为例)。

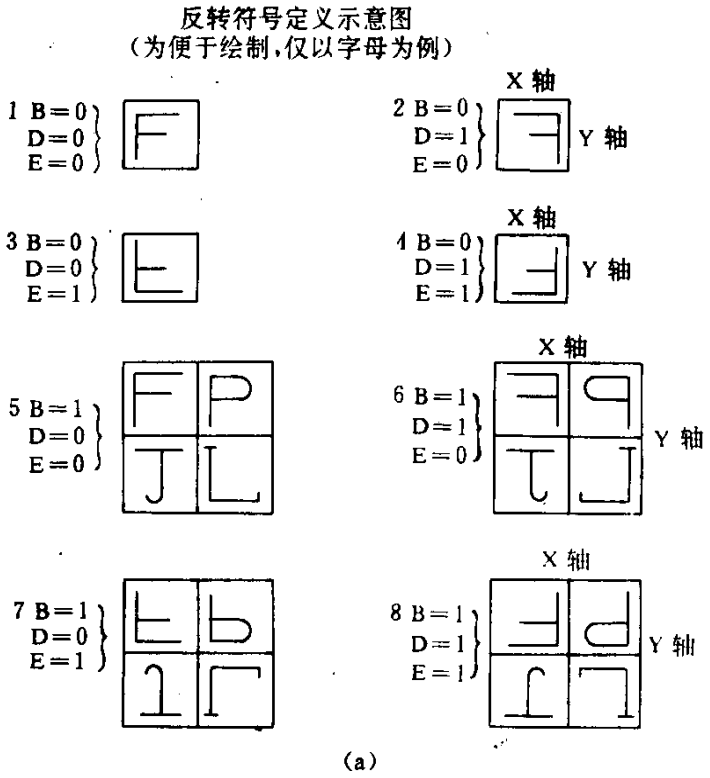


图 4—7(a)

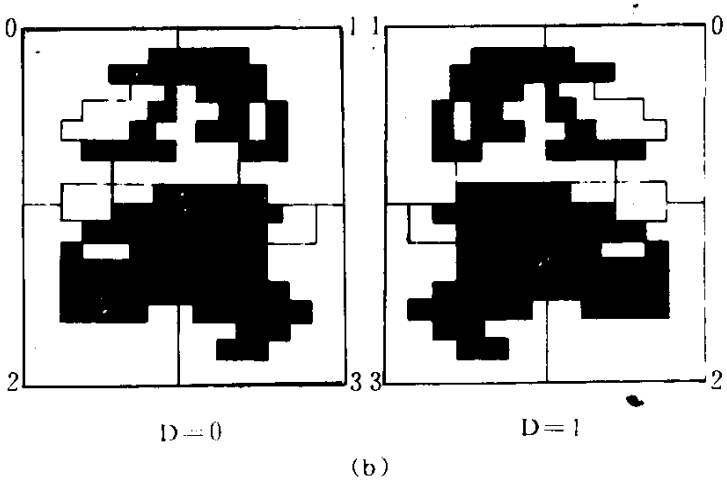


图 4—7(b)X 方向反转示意图

示例：

```
5 REM "No. 4—11"
10 SPRITE ON:CLS
20 CGSET 0,1
30 DEF SPRITE 0,(0,1,0,0,0)=CHR$(68)+CHR$(69)+CHR
   $(70)+CHR$(71)
40 DEF SPRITE 1,(0,1,0,1,0)=CHR$(29)+CHR$(28)+CHR
   $(31)+CHR$(30)
50 DEF SPRITE 2,(0,1,0,0,0)=",—./"
RUN ↙
OK
□
```

程序中30行定义了飞鱼向左飞的姿式，其显示号码为0#；40行定义了丽莎面向右走，显示编号为1#；50行定义了丽莎滑，面向左，显示编号为2#。

注意：由于现在还没有进行卡通定位，所以尽管已选定了显示SP，却仍不能显示。

4.6.3 显示定位指令(SPRITE)

格式：SPRITE n,x,y

类型：语句

简写：SP. n,x,y

功用：将由DEF SPRITE定义的SP显示于任意位置。

注释：该语句将DEF SPRITE定义的SP显示在SP面的任一位置上。

n为DEF SPRITE语句中的显示编号，欲显示哪一个SP，则n取哪一个SP的编号。n的取值范围为0～7。

x为SP面的水平方向的座标，取值范围为0～255。

y为SP面的垂直方向的座标，取值范围为0～255。

n、x、y均可取数值常数、变量或者表达式。

说明：

1. 由于显示器的差异，实际显示的SP面达不到0～255。若以0～255取可能会出现有些卡通图案显示不出来。因此，在应用中建议选取X：0～240；Y：0～220。

2. SPRITE语句可在一个画面上同时显示八个SP，超过八个以上的SP则不能显示。

3. 在同一水平方向上只能同时显示四个SP，超过四个以上的SP则不能显示。

4. 若在同一个座标上显示多个SP时，编号小的SP将显示于最前面。（即编号最小的SP显示优先度最大）。

5. 由于在DEF SPRITE语句中B＝1时SP将有四个字符大小，故由SPRITE语句决定的显示座标将作用于卡通图案的左上角。

6. 由于BG面X取值为0～27，Y取值为0～23，而SP面X取值为0～240，Y取值为0～220，所以当BG面和SP面重合时，SP在SP面的座标与在BG面的座标出现了换算问题。两面重合的情况见图4—8。

由图4—8可见，SP面大、BG面小，且两面重合不同心。其换算公式推导如下：由于一个文字为8×8点，所以应以8为比例尺。即在BG面中的一个单位，在SP中为8个单位，设：

X为SP在BG面中的x轴座标，

Y为SP在BG面中的Y轴座标。

x为SP在SP面中的X轴座标，

y为SP在SP面中的Y轴座标。

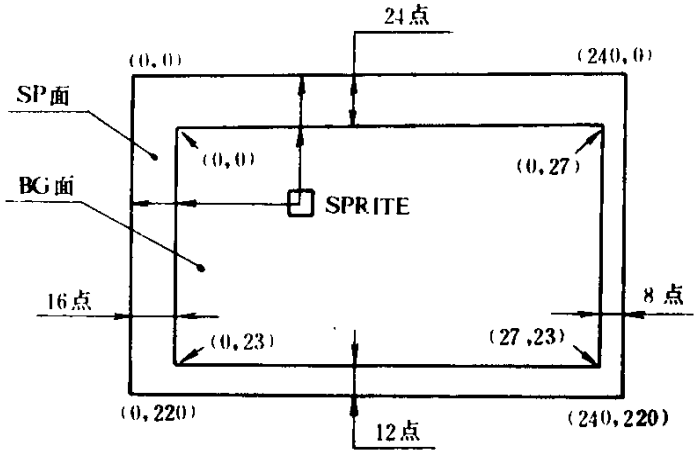


图 4—8　BG 面、SP 面重合示意图

由图 4—8 可得出

$$\begin{cases} X=8X+16 \\ Y=8Y+24 \end{cases}$$

同样可换算出

$$\begin{cases} X=(X-16)/8 \\ Y=(Y-24)/8 \end{cases}$$

注——有些键盘说明书中把 Y 轴的座标换算式写为 y＝8Y＋29，这种写法的由来是把 SP 面的起点作为 5 对待。在以后的应用中将会发现座标换算式中的补差是可变的，受影响的仅是对 SP 的作用点不同，只要正、逆换算选用同一补差则不会影响使用。

座标换算公式在游戏程序的设计中极为重要。希望大家弄通记牢。

示例：

在示例程序 No. 4—11 中加入下列语句

```
60 SPRITE 0,100,150
70 SPRITE 1,100,200
80 SPRITE 2,150,180
90 END
```

```
RUN ↙
```

则在(100,150)处显示头向左的飞鱼;在(100,200)处显示面向右的丽莎;在(150,180)处显示丽莎滑倒。

例 4.10 试编一程序,显示玛丽哥哥的七种姿式。

为压缩定义程序量,我们使用循环语句编程。

```
10 REM "No. 4—12"
20 CLS:SPRITE ON:Y=-1
30 FOR X=0 TO 27 STEP 4
40 Y=Y+1
50 DEF SPRITE Y,(0,1,0,0,0)=CHR$(X)+CHR$(X+1)+CHR
   $(X+2)+CHR$(X+3)
60 SPRITE Y,Y*25+50,Y*25+50
70 NEXT
RUN ↙
```

显示出玛丽的七种姿式。若欲显示其它 SP 只须更改 50 行中 CHF $ 括号中的数字即可,如欲显示丽莎的七种姿式,则 50 行更改如下:

```
50 DEF SPRITE Y,(0,1,0,0,0)=CHR$(X+28)+CHR$(X+29)+
   CHR$(X+30)+CHR$(X+31)
```

例 4.11 编一程序,演示 CGEN 语句的四种状态。

```
10 REM "No. 4—13"
20 CLS:SPRITE ON
30 FOR I=191 TO 255:PRINT CHR$(I);:NEXT
40 DEF SPRITE 0,(0,1,0,0,0)=CHR$(0)+CHR$(1)+CHR$(2)+
   CHR$(3)
50 SPRITE 0,100,150
60 FOR A=0 TO 3:PAUSE 300
70 CGEN A
80 LOCATE 0,23:PRINT"CGEN";A:NEXT
RUN ↙
```

循环演示 CGEN 的四种状态。读者可反复运行以上程序，仔细观察、体会 CGEN 语句的功能和变化。

4.6.4 CGEN 语句的深入分析

结合程序 No. 4—13 我们再进一步讨论 CGEN 语句的功用。在表 4—3 中我们分别从五个方面列出 CGEN 语句四种状态的屏幕变化。这是显示在屏幕上的表面现象，实质上 CGEN 指令决定了符号代码表 A(表 4—8)和符号代码表 B(表 4—2)在 BG、SP 两显示面上的符号分配。我们知道，符号代码表 B 表达了对应的数字、字母、符号和背景图案，符号代码表 A 则表达了对应的卡通图案。两张表的符号代码取值范围都是 0～255，它们是重复的。那么，对于给定的符号代码，如何决定其对应表 A 还是对应表 B 呢？这就要通过 CGEN 语句决定。见表 4—9。

表 4—9 CGEN 指令在 SP、BG 面上的分配表

n	BG 面	SP 面	分 配 情 况
0	A	A	背景面、SPRITE 面均 使用符号表 A 中的符号。
1	A	B	背景面使用表 A 的符号。 SP 面使用表 B 的符号。
2	B	A	BG 面使用表 B 的符号 SP 面使用表 A 的符号
3	B	B	BG 面、SP 面均使用表 B 中的符号。

通过表 4—9 可知，当 CGEN 语句中 n＝0 时，BG 面、SP 面均选符号表 A 中符号。因此，在 n 等于 0 时画面底背景突起，就是因为底背景在正常时为空格，由符号表 B 知，空格对应的代码是 32，而在符号表 A 中代码为 32 所对应的图形是丽莎走 2 姿式的左上部，所以在 n＝0 时底背景全部都是丽莎的半个头。同样，在正常显示时，字母取符号表 B 的字符，如字母 A 在符号表中对应的代码是 65，那么在 CGEN0 时，代码 65 在符号表 A 中对应的图案则是飞鱼 1 的一个翅膀。其它情况留给读者分析。

4.6.5 调出图形库中前 32 个图形

通过程序 No. 4—13 的运行，细心的读者可能已经发现，在 CGEN1 和 CGEN3 时，SP 变成了四个背景图案，它们正对应于背景图形库中第一行的 1～4 的四个图案。因此，我们可以通过使用 CGEN 语句调出图形库中的前 32 个图形。

例 4.12 调出图形库前 32 个图形，每八个一组，依次显示。

```
10 REM "No. 4—14"
20 CLS:SPRITE ON:CGEN 3
30 FOR A=0 TO 7
40 DEF SPRITE A,(0,1,0,0,0)=CHR$(A)
50 SPRITE A,50,A*30:NEXT
60 PAUSE 100
70 FOR I=8 TO 15
80 DEF SPRITE I-8,(1,1,0,0,0)=CHR$(I)
90 SPRITE I-8,100,(I-8)*30:NEXT
100 PAUSE 100
110 FOR I=16 TO 23
120 DEF SPRITE I-16,(2,1,0,0,0)=CHR$(I)
130 SPRITE I-16,150,(I-16)*30:NEXT
140 PAUSE 100
150 FOR I=24 TO 31
160 DEF SPRITE I-24,(3,1,0,0,0)=CHR$(I)
170 SPRITE I-24,200,(I-24)*30:NEXT
RUN ↙
```

依次显示 32 个背景图形，每 8 个图形变换一次颜色。上程序中，我们把背景图形作为 SP 对待，用 SP 控制语句作用于背景图形，在 CGEN 语句的支持下达到了目的。同样，《符号代码表 B》中的数字、字母、符号也可以作为 SP 调用。后文将看到，各种字符、图案还可以用卡通运动控制语句使其向任意方向运动。

为便于读者理解接受，程序 No. 4—14 使用重复语句较多，我

们可再加一重循环语句，构成双重循环以简化程序（见 No. 4—15）。

```
10 REM "No. 4--15"
20 CLS:SPRITE ON:CGEN 3:X=-1
30 FOR A=0 TO 31 STEP 8
40 X=X+1
50 FOR I=0 TO 7
60 DEF SPRITE I,(X,1,0,0,0)=CHR$(A+I)
70 SPRITE I,50+50*X,I*30:NEXT
80 PAUSE 100:NEXT
RUN ↙
```

演示效果与 No. 4—14 相同。

程序 No. 4—15 仍嫌烦琐，还可进一步简化见程序 No. 4—16。

```
5 REM"No. 4—16"
10 CLS:SPRITE ON:CGEN 3
20 FOR K=0 T0 3:FOR A=0 TO 7
30 DEF SPRITE A,(K,1,0,0,0)=CHR$(A+k*8)
40 SPRITE A,(K+1)*50,A*30:NEXT
50 PAUSE 100:NEXT
```

这可能是解决例 4.12 问题的最简程序了。

4.6.6 消显示指令(SPRITE n)

格式：SPRITE n

类型：语句

简写：SP. n

功用：令已显示的 SP 在画面上消失。

注释：n 为由 DEF SPRITE 语句定义的显示卡通编号，取值为 0～7。它可以是数值常数，可以是变量，也可以是表达式。

示例：

```
10 REM "SPRITE n"
```

```
20 CLS:SPRITE ON:Y=-1
30 FOR I=0 TO 27 STEP 4:Y=Y+1
40 DEF SPRITE Y,(Y/2,1,0,0,0)=CHR$(I)+CHR$(X+1)+CHR
   $(X+2)+CHR$(X+3)
50 SPRITE Y,Y*25+50,Y*25+50
60 NEXT
70 FOR I=0 TO 6
80 PAUST 100:SPRITE I
90 NEXT
100 END
RUN ↙
```

屏幕上依次打印玛莉哥哥的七种姿式，然后延时一段时间，消失一种姿式；再延时一段时间又消失一种姿式；直至七个 SP 全部消失。

可见 SPRITE n 指令可以有选择地消失 SP。

4.6.7 关显示指令(SPRITE OFF)

格式：SPRITE OFF

类型：语句或命令

简写：SP. OF.

功用：关闭 SP 显示开关。

SPRITE OFF 指令可令显示于画面上的所有 SP 全部消失。

注释：SPRITE OFF 为关闭 SP 的意思。

示例：

```
10 REM "SPRITE OFF"
20 CLS:SPRITE ON
30 DEF SPRITE 0,(0,1,0,0,0)="DEFG"
40 DEF SPRITE 1,(0,1,0,0,0)="4567"
50 SPRITE 0,100,100
60 SPRITE 1,150,150
70 PAUSE 300:PRINT "USE SPRITE OFF!"
```

```
80 SPRITE OFF
90 LOCATE 0,22:END
RUN ↙
```

首先显示飞鱼和丽莎，延时一段时间后，打印“USE SPRITE OFF!”(使用关显示指令)后，飞鱼和丽莎消失。

说明:SPRITE OFF 和 SPRITE n 的区别：

1. SPRITE n 有选择地消失显示的 SP，而 SPRITE OFF 则令所有显示 SP 消失。

2. SPRITE n 只能令 DEF SPRITE 语句定义的 SP 消失，而 SPRITE OFF 还能消失由 DEF MOVE 语句定义的 SP。

4.7 卡通运动控制指令

显示指令只能令 SP 显示于画面之上，不能使其运动，而在游戏中最需要的是卡通运动。本节重点介绍卡通的运动控制指令。

令卡通运动，首先要选择运动卡通的类型，选定运动的方向、速度等，这就要使用定义运动语句(DEF MOVE)；运动前要决定卡通的运动起点位置，因而要使用定位语句(POSITION)确定起点座标；运动中要随时掌握卡通的运行情况以决定游戏的进程，从而要使用卡通目前方位测试语句(XPOS、YPOS)，运动方向测试语句(VCT)，运动完成测试语句(MOVE(n))和是否与其它卡通相遇碰撞测试语句(CRASH)；在游戏中卡通碰到障碍物要停止运动，卡通被枪弹击中要消失，从而还要使用卡通停动和消失语句(CUT、ERA、CAN)等等。

总之，卡通运动控制指令是游戏程序设计的关键指令。除 POSITION 外，其余 10 条指令都是 F BASIC 所独有的指令。因此，在各指令的介绍上，尽量做到详尽清楚，以利读者掌握使用，顺利编制游戏程序。

4.7.1 定义 SP 动作语句(DEF MOVE)

格式:DEF MOVE(n)=SPRITE(A,B,C,D,E,F)

类型:语句

简写:DE. M.

功用:定义卡通图案的动作。

通过 DEF MOVE 语句,确定动作的编号、运动卡通的种类、运动方向、运动速度、一次运动量、相对于背景的显示优先度和运动卡通的配色号码。

注释:

1. DEF MOVE 取自 Define move,是定义运动的意思。

2. n 为动作编号,取值范围 0～7。(注:这里的 n 与 DEF SPRITE 语句中的 n 意义不同,注意不要混淆。)F BASIC 允许一次定义 8 个动作,即在同一画面最多允许有 8 个 SP 运动。

3. A 为 SP 的种类编号。它对应于表 4—7 中的 16 种 SP 编号。A 的取值范围为 0～15。

4. B 为动作的运动方向。取值范围为 0～8。运动卡通可以向八个方向运动,每一个运动方向与一个代码对应,对应情况见表 4—10 和图 4—9。

表 4—10 卡通运动方向代码表

B	运动方向	B	运动方向
0	静止	5	向下
1	向上	6	左下
2	右上	7	向左
3	向右	8	左上
4	右下		

5. C 为运动速度,取值范围为 1～255。C=1 时速度最快,C=255 时速度最慢。C 的取值与速度的对应关系为

V=60/C(点/秒)

速度 V 的计量单位为"点/秒",即每秒移动 n 点的意思。如:

C=1　　则 1 秒种移动 60 点，V=60

C=3　　则 3 秒种移动 60 点，V=20

C=60　　则 60 秒钟移动 60 点，V=1

即，C=1 时每秒钟移动 60 点；

C=3 时每秒钟移动 20 点；

C=60 时每秒钟移动 1 点；

注意：C 等于 0 时则每隔 256 个单位动一次。

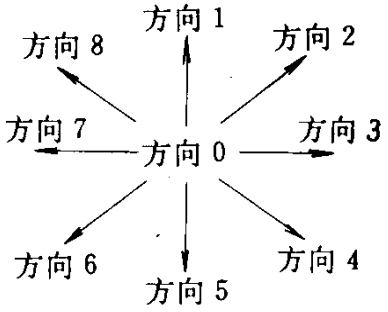


图 4－9　SP 运动方向示意图

6. D 为一次移动量，取值范围为 1～255。一次移动量是指选定动作一次移动的距离。D 的取值与移动距离(S)的对应关系为：

S=2×D　（点）　　（水平、垂直方向）

S 的计量单位为“点”。如：

D=1　　则移动 2 点；

D=100　　则移动 200 点。

在倾斜方向时移动距离与 D 的对应关系为：

$S=2\times\sqrt{2}\times D$　（点）　　（倾斜方向）

注意：D 不能等于 0，否则，该动作不能显示。

7. E 为显示优先度，取值为 0、1。

E=0　则从背景前面显示 SP；

E=1　则从背景后面显示 SP。

当 SP 在背景后面显示时，若为网状背景则隐约可见 SP；若为

细密背景则看不见 SP。

8. F 为配色号码，取值范围为 0～3。F 的取值是由 CGSET 语句选定 SP 调色板中的颜色组合代码。

9. DEF MOVE 语句中的所有元素都可取数值常数、变量或表达式。

示例 1：

```
10 DEF MOVE(0)=SPRITE(10,1,1,100,0,0)
```

10 句定义了 0# 动作为 10 号 SP 爆炸以每秒 60 点的速度向上移动 200 点。

示例 2：

```
20 DEF MOVE(3)=SPRITE(4,2,2,35,1,1)
```

20 句定义了 3# 动作为 4 号 SP 企鹅，它以每秒 30 点的速度向右上运动 70 点，且在背景后面显示、取 1 号 SP 调色板中的第二组颜色组合。

另外，在 DEF MOVE 语句中的运动方向 B、显示优先度 E、配色号码 F 可以省略。省略 B 后，计算机自动选定运动方向为 4；省略 E 时则自动设定 E＝0；省略 F 时则自动设定 F＝0，并使 SP 从(0,0)处开始运动。省略时要注意 B、E、F 可以省略，但其分隔符“,”要保留。如：

DEF MOVE(0)＝SPRITE(2,,1,20,,)若同时省略 E、F 时，可把最后的两个分隔符“,”去掉。如上语句也可写为：

DEF MOVE(0)＝SPRITE(2,,1,20)两语句形式虽不同，效果是一样的，都定义了 2# 卡通以每秒 60 点的速度向右下方移动 40 点。

4.7.2 定位语句(POSITION)

动作定义后，要做的工作就是确定卡通运动的起点座标，然后才能启动使其运动。确定运动起点座标的过程叫定位，定位则通过 POSITION 语句实现。

格式:POSITION n,X,Y

类型:语句

简写:POS.

功用:规定 SP 动作前的座标。

注释:POSITION 是定位的意思。

n 为 DEF MOVE 语句定义的动作编号,取值范围为 0～7。

X 为动作前 SP 在水平方向的座标,取值范围为 0～255。(实用中一般取 0～240)。

Y 为动作前 SP 在垂直方向的座标,取值范围为 0～255。(实用中一般取 5～220)。

当程序中未指定动作座标时,则系统自动约定为 X＝120、Y＝120。

示例:

```
POSION 1,100,200
```

定义了 1# 动作的起始座标为(100,200)。

4.7.3 启动指令(MOVE)

一个动作必须通过 DEF MOVE 定义。由 POSITION 确定运动起点座标,最后经启动指令(MOVE)启动后才能运动。

格式:MOVE n_0,n_1,n_2,n_3,n_4,n_5,n_6,n_7

类型:语句

简写:M.

功用:启动定义的卡通图案开始动作。

注释:MOVE 为动作、运动的意思。

n_0～n_7 为启动动作的编号。它们由 DEF MOVE 指定。

MOVE 语句可以一次启动八个 SP 开始运动,也可以只启动一个。如:

```
MOVE 0
MOVE 0,1,3
```

```
MOVE 6,7,5,0,2,3,4
……
```

示例：

```
10 REM "MOVE"
20 CLS:SPRITE ON
30 DEF MOVE(0)=SPRITE(2,3,3,100,0,0)
40 POSITION 0,100,50
50 MOVE 0
60 END
RUN ↙
```

显示一个苍蝇在座标为(100,50)处开始向右飞行，最后停留在屏幕的左侧。

通过上例的运行，我们有两点发现：

1. 当苍蝇向右飞行到屏幕边缘时，即在这一边消失，犹如向右飞出屏幕；当在右边消失后则立即在屏幕左边出现，犹如从左边飞入屏幕。由此说明，对 SP 的运动来说，屏幕犹如一个展开的球体，SP 在球表面运动，所以没有边缘。

2. 当苍蝇开始飞行时，屏幕上就打印出程序执行完毕的提示信息"OK"，同时出现闪烁光标。这说明程序运行已经结束，而这时的 SP 却仍然继续飞行，直到飞行完定义的 200 点的运动量才停止。这告诉我们一个结论：当启动指令 MOVE 被执行后，即使程序运行结束，SP 依然会依 DEF MOVE 语句所定义的运动、方向、速度和移动量继续运动，直至动作完了为止。

删去上例中的 40 行，重新运行程序，可以看到这时的苍蝇则从座标为(120,120)处开始运动。从而说明了当未指定 POSITION 语句时，计算机自动设定默认值。

例 4.13 试编一程序，令 SP 运动起点座标随机变化。

```
10 REM "No. 4—17"
20 CLS:SPRITE ON
```

```
30 DEF MOVE(0)=SPRITE(14,3,2,10,1,2)
40 X=RND(256):Y=RND(240)
50 PRINT "X,Y=";X;",";Y
60 POSITION 0,X,Y
70 MOVE 0
80 PAUSE 80:GOTO 20
RUN ↙
```

显示螃蟹在不同的位置向右运动，同时屏幕左侧打印出每次运动的起点座标(X,Y)。

例 4.14 试编程序令太空船的运动起点和运动方向随机变化。

```
10 REM "No. 4—18"
20 CLS:SPRITE ON
30 I=RND(9):IF I=0 THEN 30
40 X=RND(240):Y=RND(220)
50 DEF MOVE(0)=SPRITE(9,I,1,100,0,0)
60 POSITION 0,X,Y:MOVE 0
70 PAUSE 300:GOTO 30
```

例 4.15 试编一程序使画面上显示八个卡通图案分别向八个方向运动。运动起点座标随机变化，运动反复进行。

```
10 REM "No. 4—19"
20 CLS:SPRITE ON
30 FOR I=0 TO 7
40 DEF MOVE(I)=SPRITE(I,I+1,I+1,100)
50 X=RND(240):Y=RND(220)
60 POSITION I,X,Y
70 MOVE I:NEXT
80 PAUSE 500:GOTO 30
RUN ↙
```

显示 0～7 号八个卡通从不同的起点，以不同的速度，向八个方向反复运动。

若要第 8～15 号卡通运动，可将 40 行中的 I 改为 8+I 即可。即

```
40 DEF MOVE(I)=SPRITE(8+I,I+1,I+1,100)
```

若使八个 SP 以不同的色彩出现，可把 30 行更改如下：

```
40 DEF MOVE(I)=SPRITE(8+I,I+1,I+1,100,0,I/2)
```

以上几例程序的 SP 起点座标都是随机产生的，从而使 SP 的运行路线不连贯。怎样使运行路线连贯呢？请看例 4.16。

例 4.16 试编一程序，令 SP 的运行路线连贯。

```
10 REM "No. 4—20"
20 CLS:SPRITE ON
30 DEF MOVE(0)=SPRITE(0,3,1,30,0,0)
40 POSITION 0,100,100
50 MOVE 0
60 PAUSE 100:GOTO 50
RUN ↙
```

运行情况是：SP 运行 60 点后，暂停几秒钟又开始运动 60 点，运行路线是连续的，达到了目的要求。但是，SP 每运行一次都要暂停一会，运行动作不连贯。不连贯的原因，可能是暂停时间太长。是否缩短暂停时间可以解决呢？现在试试看。把暂停时间由 100 改为 80，不行；改为 60，也不行；改为 30，仍然不行。可见解决动作连贯问题并不简单。分析起来，主要是我们选择的暂停时间与 SP 的运行时间没有完全重合所致。若暂停时间大于 SP 的运行时间，则 SP 要暂停一会儿才开始运行；若暂停时间小于 SP 的运行时间，则暂停时间已到而 SP 的动作还没有停止，MOVE 语句不起作用，从而又转入第二次暂停；只有暂停时间结束恰好 SP 动作也结束，才能达到同步，实现动作连贯。实际上，这种情况很难凑巧。SP 动作结束判断语句 MOVE(n)则很好地解决了这一问题。

4.7.4 动作完成测试语句(MOVE(n))

格式：MOVE(n)

类型:函数

简写:M.(n)

功用:判断 SP 动作是否完成。

注释:n 是 DEF MOVE 所选定的 SP 的动作号码,取值范围为 0～7。

当定义的 SP 经 MOVE 语句启动后,MOVE(n)函数则对应指定的功作被赋给新值。若动作在执行中则 MOVE(n)＝－1;若动作已经完成则 MOVE(n)＝0。

示例

```
10 REM "MOVE(n)"
20 CLS:SPRITE ON:CGSET 1,0
30 DEF MOVE(0)=SPRITE(0,3,1,150,0,0)
40 MOVE 0
50 IF MOVE(0)=-1 THEN PRINT "MOVE(0)="MOVE(0):GOTO 50
60 PRINT"MOVE(0)="MOVE(0)
70 END
RUN ↙
```

显示玛丽哥哥从屏幕中央向右运动,同时打印出 MOVE(0)＝－1。只要玛丽运动不停止,一直打印 MOVE(0)＝－1,直到运动停止则打印出 MOVE(0)＝0,停机。

现在,我们再进行一次 SP 动作连续的试验。

```
10 REM "CONTINUAL MOVE"
20 CLS:SPRITE ON:CGSET 1,1
30 DEF MOVE(0)=SPRITE(0,3,1,50,0,0)
40 POSITION 0,50,100:MOVE 0
50 IF MOVE(0)=0 THEN MOVE 0
60 GOTO 50
RUN ↙
```

现在 SP 的动作连续了。

4.7.5 测试 SP 水平座标函数(XPOS)

格式:XPOS(n)

类型:函数

简写:XP.(n)

功用:求出 SP 动作前的水平座标值。

注释:XPOS 取自于 X AXIS POSITION,为 X 轴位置的意思。

n 为 DEF MOVE 语句定义的动作编号,取值范围为 0～7,可以是变量或表达式。

示例:

```
10 REM "XPOS"
20 CLS:SPRITE ON
30 DEF MOVE(1)=SPRITE(0,3,2,10)
40 MOVE 1
50 LOCATE 8,20:PRINT"␣␣␣"
60 LOCATE 0,20:PRINT "XPOS(1)="XPOS(1)
70 PAUSE 50:GOTO 40
RUN ↙
```

显示玛莉向右运动,屏幕左下角间歇打印 XPOS(1)的函数值,即玛莉在 X 方向的座标。

4.7.6 测试 SP 垂直座标函数(YPOS)

格式:YPOS(n)

类型:函数

简写:YP.

功用:求出 SP 动作前的垂直座标值。

注释:YPOS 取自于 Y AXIS POSITION,为 Y 轴位置的意思。

n 为 DEF MOVE 语句定义的动作编号,取值范围为 0～7,可以是变量或表达式。

示例:

```
10 REM "YPOS"
```

```
20 CLS:SPRITE ON:CGSET 0,0
30 DEF MOVE(3)=SPRITE(15,2,2,10,0,1)
40 MOVE 3
50 LOCATE 8,20:PRINT" "
60 LOCATE 8,21:PRINT" "
70 LOCATE 0,20:PRINT"XPOS(3)="XPOS(3)
80 LOCATE 0,21:PRINT"YPOS(3)="YPOS(3)
90 PAUSE 50:GOTO 40
RUN ↙
```

间歇打印飞鸟的 X、Y 座标。

例 4.17 试编一演示程序，令玛丽连续向八个方向运动。

```
10 REM "No. 4—21"
20 FOR I=1 TO 8
30 DEF MOVE(4)=SPRITE(0,I,1,100,0,0)
40 POSITION 4,X,Y
50 MOVE 4:PAUSE 300
60 X=XPOS(4):Y=YPOS(4)
70 NEXT
RUN ↙
```

显示玛丽从座标(50,50)开始连续向八个方向运动。

例 4.18 编一令玛莉哥哥走八边形的程序。

```
10 REM "No. 4—22"
20 CLS:SPRITE ON
30 LOCATE 9,9:PRINT"Ma Li Ge Ge"
40 LOCATE 6,10:PRINT"Zou Ba Bian Xing"
50 X=35:Y=130
60 FOR I=0 TO 7
70 DEF MOVE(0)=SPRITE(0,I+1,1,30)
80 POSITION 0,X,Y
90 MOVE 0
```

```
100 IF MOVE(0)=-1 THEN 100
110 X=XPOS(0):Y=YPOS(0)
120 NEXT
RUN ↙
```

首先打印出“玛莉哥哥走八边形”的汉语拼音，随后玛莉从座标(35,130)开始向上走 60 点；在(35,70)处向右上走 84 点；在(95,10)向右走 60 点；在(155,10)处开始向右下走 60 点；在(215,70)处向下走 60 点；在(215,130)处开始向左下走 84 点；在(155,190)处向左走 60 点；在(95,190)处向左上走 84 点；最后在(35,130)处停止，与出发点重合，其运动轨迹如图 4—10。

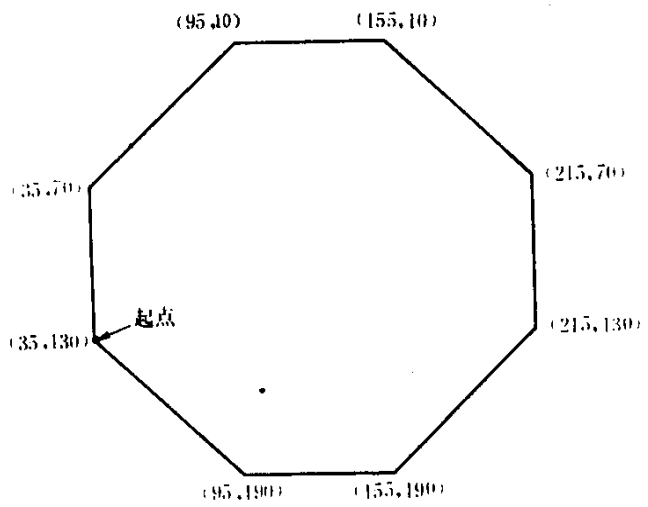


图 4—10 玛莉运行轨迹

4.7.7 停止 SP 运动指令(CUT)

格式：CUT n_0,n_1,n_2,n_3,n_4,n_5,n_6,n_7

类型：语句

简写：CU.

功用：令 MOVE 语句启动的 SP 在运动中停止。

CUT 指令停止运动的 SP，可用 MOVE 再启动。再启动后，继续

运行上一次未执行完的移动量，直到运行完定义的移动量才停止。

注释：CUT 取自于 cut move，是停止运动的意思。

n_0～n_7 是 DEF MOVE 语句定义的 SP 的动作号码。CUT 语句可以一次停止 8 个 SP 的动作，也可以只停止一个。如：

```
CUT 1
CUT 0,1,3,7
CUT 7,4,2,5,6,
```

都是合法的。

示例：

```
10 REM "CUT"
20 CLS:SPRITE ON
30 DEF MOVE(7)=SPRITE(1,3,2,255,0,1)
40 POSITION 7,30,120
50 MOVE 7:PAUSE 50
60 LOCATE 0,0:PRINT"USED CUT"
70 CUT 7:PAUSE 50
80 LOCATE 0,0:PRINT"USED MOVE"
90 GOTO 50
RUN ↙
```

首先显示丽莎从(30,120)处开始向右运动，随后打印：

```
USED CUT          (使用 CUT 语句使 SP 停止)
```

丽莎停止运动。暂停一会，又打印

```
USED MOVE         (使用 MOVE 语句再启动)
```

丽莎又开始运动。现在运动是在上一次停止处直接运动。运动一段距离，又打印

```
USED CUT
......
```

通过丽莎的停止、运动，运动又停止的重复过程可深刻地认识到 CUT 语句的作用。

4.7.8 令 SP 消失指令(ERA)

格式:ERA n_0,n_1,n_2,n_3,n_4,n_5,n_6,n_7

类型:语句

简写:ER.

功用:令由 MOVE 启动运动的 SP 停止动作,并从画面上消失。

注释:ERA 取自于 Erase,是擦除的意思。

n_0～n_7 是 DEF MOVE 语句定义的 SP 的动作号码。ERA 语句可以一次停止并消失 8 个 SP,也可以只消失一个 SP。如:

```
ERA 0
ERA 3,4,5,
ERA 0,7,6,5,3,4,2
```

都是合法的。

ERA 语句还可以令由 CUT 语句停止运动的 SP 消失。

被 ERA 语句作用消失的 SP,可用 MOVE 语句令其重新显示,并继续运动消失前未完的移动量。重新显示的位置座标仍是消失前的座标。

示例:

```
10 REM "ERA"
20 CLS:SPRITE ON:CGSET 1,0
30 DEF MOVE(0)=SPRITE(0,3,2,255,0,1)
40 POSITION 0,30,150
50 MOVE 0:PAUSE 50
60 LOCATE 0,0:PRINT "USED CUT"
70 CUT 0:PAUSE 50
80 LOCATE 0,0:PRINT "USED ERA"
90 ERA 0:PAUSE 50
100 LOCATE 0,0:PRINT"USED MOVE"
110 GOTO 50
```

RUN ↙

显示玛莉运行一段距离，打印

USED CUT

停止运动。又过一段时间，打印

USED ERA

SP 消失。又延时一段时间，打印

USED MOVE

消失的 SP 又从消失处显示并继续运动。

……。

4.7.9 解除 SP 定义指令(CAN)

格式：CAN n_0，n_1，n_2，n_3，n_4，n_5，n_6，n_7

类型：语句

功用：解除 DEF MOVE 语句定义的 SP。

CAN 语句可使 MOVE 语句启动的卡通在移动中永远消失，即使再使用 MOVE 指令也不能启动显示。

注释：CAN 取自于 Cancel，是取消的意思。

n_0～n_7 为 DEF MOVE 语句定义的 SP 动作号码。CAN 语句可一次解除 8 个 SP 的定义。也可以只解除 1 个或几个 SP 的定义。如

CAN 0

CAN 3,2,1,7

CAN 0,1,2,3,4,5,6,7

都是合法的。

被 CAN 语句作用的动作只有经 DEF MOVE 语句重新定义后，才能用 MOVE 语句使其重新显示和运动。

说明：CUT、ERA、CAN 三个语句的不同点在于：CUT 语句只令 SP 运动停止，不能令 SP 消失；ERA 语句可令 SP 运动停止并从屏幕上消失；CAN 语句则令 SP 运动停止且令 SP 永远消失。CUT、ERA 语句作用于 SP 后，可使用 MOVE 语句重新启动运动，而 CAN 语句作用的 SP 则永远消失，只能重新定义后才能运动。

例 4.19 编程演示 CUT、ERA、CAN 语句功能的区别。

```
10 REM "No. 4—23 CUT, ERA AND CAN"
20 CLS:SPRITE ON:CGSET 0,1
30 FOR I=0 TO 2
40 DEF MOVE(I)=SPRITE(I,3,1,100,0,0)
50 POSITION I,10,I*50+70
60 MOVE I:NEXT
70 PAUSE 100
80 CUT 0:PRINT "CUT 0":PAUSE 100
90 ERA 1:PRINT "ERA 1":PAUSE 100
100 CAN 2:PRINT "CAN 2":PAUSE 100
110 MOVE 0,1,2:PRINT "MOVE 0,1,2"
RUN ↙
```

显示 3 个 SP:0 号玛莉哥哥,1 号丽莎小姐,2 号飞鸟自左向右运动。运动大约 2 秒钟玛莉停止运动,屏幕左上角打印 CUT 0;又过两秒钟,丽莎消失,打印 ERA 1;再两秒钟后,飞鸟消失,左上角显示 CAN 2;两秒钟后,丽莎又出现与玛莉同时运动,左上角显示 MOVE 0,1,2;这时,飞鸟已不再出现。这充分说明了三个语句的不同点。

程序 No. 4—23 可反复运行多观察几次,也可以补入

```
120 PAUSE 100:GOTO 30
```

使程序自动反复运行。

4.7.10 SP 碰撞测试指令(CRASH)

在游戏程序的设计中,经常要判断一个 SP 是否与其它 SP 相遇——碰撞。若相遇则作某种处理。如在采蘑菇游戏中,当玛莉与红蘑菇相遇时,玛莉的身体要比以前增大一倍(即玛莉吃掉红蘑菇,红蘑菇消失,玛莉长大一倍)。这就要在游戏中随时判断玛莉是否与红蘑菇相遇,若相遇则转去执行红蘑菇消失和玛丽长大子程序,然后返回。若不相遇,则继续执行后续程序。在 F BASIC 语言

中判断 SP 碰撞的指令是 CRASH 函数。

格式:CRASH(n)

类型:函数

简写:CR.(n)

功用:给出与被测试 SP 碰撞的 SP 的动作号码。

注释:CRASH 是碰撞的意思。

n 为被测试 SP 的动作号码。取值范围为 0～7。n 可以是常数、变量或表达式。

若被测试 SP 的动作号码为 0,与其碰撞的 SP 的动作号码为 2,则有 CRASH(0)=2。

若未相撞则 CRASH(0)=−1。

例 4.20 令 0～7 号 SP 随机运动,分别测试八个 SP 与其它 SP 相撞的情况,打印出两相撞 SP 的动作号码。

```
10 REM "No. 4—24 CRASH"
20 CLS:SPRITE ON
30 FOR I=0 TO 7
40 DEF MOVE(I)=SPRIET(I,I+1,I+1,255,0,0)
50 X=RND(170):Y=RND(150)
60 POSITION I,X,Y
70 MOVE I
80 A=CRASH(I):IF A=-1 THEN 100
90 PRINT I"# CRASH"A"#"
100 NEXT
110 PAUSE 300:GOTO 30
RUN ↙
```

显示八个 SP 在屏幕上随机运动,并能断续打印出如下内容:

```
5# CRASH 3#          (表示 5#SP 与 3#SP 相撞)
2# CRASH 4#          (表示 2#SP 与 4#SP 相撞)
3# CRASH 5#          ……
```

```
6# CRASH 4#
.....
```

上述程序随机测试两卡通相撞的情况。我们可以固定测试一个SP与其它SP的相撞情况，如测试7号动作太空堡垒，可改程序No. 4—24如下：

```
10 REM "No. 4—25"
20 CLS:SPRITE ON
30 FOR I=0 TO 7
40 DEF MOVE(I)=SPRITE(I,I+1,8-I,255,0,0)
50 X=RND(170):Y=RND(150)
60 POSITION I,X,Y
70 MOVE I
80 NEXT
90 A=CRASH(7)
100 PRINT "7# CRASH"A"#":IF MOVE(7)=-1 THEN 90
110 GOTO 30
120 END
RUN ↙
```

现在只打印与7号SP相撞的动作号码。如

```
7# CRASH 0#
7# CRASH 3#
7# CRASH -1#
.....
```

"-1#"是表示没有SP与7#相撞。可从屏幕上清楚地看到，CRASH值的变化与碰撞情况一一对应。

4.7.11 SP运动方向测试指令(VCT)

格式：VCT(n)

类型：函数

简写：VC.(n)

功用：测试SP的运动方向，给出方向代码。

注释：VCT 取自 Verify Cartoon 是检验卡通的意思。

n 为 DEF MOVE 语句定义的 SP 的动作号码。取值范围 0～7，它可以是常数、变量或表达式。

示例：

```
10 REM "VCT"
20 CLS:SPRITE ON
30 DEF MOVE(7)=SPRITE(6,RND(9),2,10)
40 POSITION 7,RND(170),RND(150)
50 MOVE 7
60 A=VCT(7):LOCATE 0,0:PRINT "7# VCT="A
70 IF MOVE(7)=-1 THEN 70
80 GOTO 30
90 END
```

RUN ↙ 则打印

7# VCT=4 或

7# VCT=2 或

7# VCT=1

……

4.8 控制 BG 画面指令

通过以上的学习，我们已初步掌握了卡通图案的定义、显示和控制手段，只要配上适当的背景就可设计出简单的游戏程序。

我们已经发现，编制程序绘图，程序量大、占用内存多。实际上，我们的学习机已设计有最简便的绘图方法——键控绘图。这也正是我们的学习机优于其它微机的特点之一。通过对话方式进入 BG GRAPHIC 状态后，我们就可以随意选择背景图形库中的 104 种图案，使用极少的几个键，绘制出满意的图形。使用时，只要在程序的开始部分添一简单的语句，就可调出已绘好的画面，从而达到

了不占有效内存而绘制复杂图形的目的。调出已绘图形的指令叫VIEW，其格式如下：

格式：VIEW

类型：语句或命令

简写：V.

功用：调出 BG GRAPHIC 画面。

在 BASIC 程序的执行中，使用 VIEW 语句可随时调出已绘好的背景画面。

注释：VIEW 是观看、观察的意思。

VIEW 的实质就是复制一份与 BG GRAPHIC 相同的画面。

演示步骤：

1. 通过对话选择主菜单中的第二项“BG GRAPHIC”进入键控绘图画面。

2. 按键控绘图步骤绘制满意的图案。

3. 绘图完毕，按 Esc、Stop 键返回主菜单，选择第一项进入对话 BASIC。

4. 编制 BASIC 程序。

5. 编程完毕，运行修改，一切正常后在程序的 CLS 语句之后加入“VIEW”。

6. 现在运行程序，即可立即调出绘好的背景图案。并继续往下执行程序。

通过使用，我们发现这样一个问题：利用背景绘图的方法只适用对话 BASIC，而不能在直调 BASIC 中使用，对话 BASIC 仅有 4K 内存供用户使用(占直调 BASIC 的 1/2)，程序稍大一点就可能产生溢出而与系统程序相冲突。并且对话 BASIC 所能识别的指令较少，处理能力较差。由于直调 BASIC 有 8K 内存供用户使用，且能识别的指令较多，因此我们更希望使用直调 BASIC 编制程序。

那么，如何在直调 BASIC 状态利用背景绘图呢？F BASIC 语言

系统还有一条鲜为人知的返回 BG GRAPHIC 状态指令——BG-TOOL。格式如下：

格式：BGTOOL

类型：语句或命令

简写：BG.

功用：进入 BG GRAPHIC 绘图状态。

使用 BGTOOL 指令进入 BG 绘图状态后可以对话进入的方法进行键控绘图。

注释：BGTOOL 取自 Back to Graphic Tool，是返回背景绘图的意思。

使用方法：

1. 作为命令使用：在直调 BASIC 状态下，键入 BGTOOL 并回车，就进入键控绘图状态。绘图完毕后，按 ESC、STOP 键则返回直调 BASIC 状态。使用时，在程序的适当位置加入 VIEW，即可调出已绘好的图形。

2. 作为语句使用：可在程序中加一菜单选择程序。若选择键控绘图则转 BGTOOL 所在的语句行，从而进入 BG GRAPHIC 状态绘图。返回方法与第 1 相同。若不选择键控绘图，则跳过 BGTOOL 所在语句行，执行下面的语句。

示例：

```
10 REM "BGTOOL—VIEW"
20 CLS:SPRITE ON
30 INPUT "PLEASE SELECT 1—BGTOOL,2—VIEW";A
40 ON A GOTO 50,60
50 BGTOOL
60 VIEW
70 ……
RUN          ↙      显示
PLEASE SELECT 1—BGTOOL,2—VIEW? □
```

键入 1 ↙

进入 BG GRAPHIC 键控绘图画面。返回后重新运行程序。

键入 2 ↙

调出 BG 画面，程序继续向下进行。

例 4.21 画一方框，框内加两条斜线，形成一不规则六边形，在六边形内显示丽莎的七种姿式。

```
10 REM "No. 4—27"
20 CLS:SPRITE ON:Y=-1
30 BGTOOL
40 FOR X=0 TO 27 STEP 4:Y=Y+1
50 DEF SPRITE Y,(0,1,0,0,0)=CHR$(X+28)+CHR$(X+29)+
   CHR$(X+30)+XHR$(X+31)
60 SPRITE Y,Y*25+40,Y*20+40
70 NEXT
80 LOCATE 0,22
90 END
RUN ↙
```

调出了 BG GRAPHIC 状态，这时可按键控绘图的方法，选择你喜欢的图形和色彩绘制图 4—11 的图案。然后键入 Esc 和 Stop 返回直调 BASIC 状态，把 30 行的 BGTOOL 改为 VIEW，重新运行程序，则立即打印出背景图案如图 4—11，丽莎的七种姿式显示在六边形内。

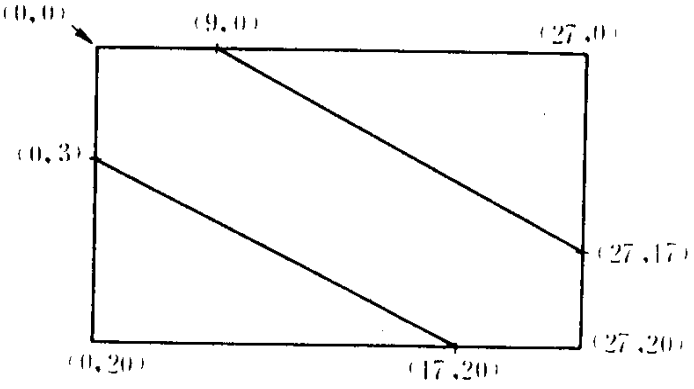


图 4—11

4.9 操纵器控制指令

玩游戏的过程就是用操纵器控制游戏中的主人翁动作的过程。F BASIC 语言提供了两条操纵器控制指令:STICK 和 STRIG。使用这两条指令可灵活地控制 SP 完成跳跃、射击、向各个方向运动等动作。

4.9.1 SP 运动方向控制指令(STICK)

格式:STICK(x)

类型:函数

简写:STI.

功用:控制 SP 的运动方向。

STICK 函数对应操纵器的每一个方向都有一个键值,根据键值可间接地确定 SP 的运动方向。

注释:STICK 是指挥棒、操纵杆的意思。

X 为选定操纵器的代码,取值 0、1。X=0 表示选 I 号操纵器;X=1 表示选 Ⅱ 号操纵器。X 可以取数值、变量和表达式。

操纵器共有八个方向,每个方向对应的键值如下图:

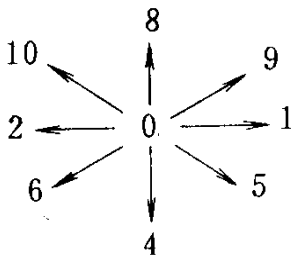


图 4-12 操纵器方向键值

示例

```
10 REM "STICK"
20 CLS
```

```
30 S=STICK(0)
40 IF S=1 THEN PRINT "RIGHT"
50 IF S=2 THEN PRINT "LEFT"
60 IF S=4 THEN PRINT "DOWN"
70 IF S=8 THEN PRINT "UP"
80 IF S=5 THEN PRINT "DOWN RIGHT"
90 IF S=6 THEN PRINT "DOWN LEFT"
100 IF S=9 THEN PRINT "UP RIGHT"
110 IF S=10 THEN PRINT "UP LEFT"
120 GOTO 30
130 END
```

示例程序选Ⅰ号操纵器。

```
RUN ↙ 按向左键,则打印
LEFT 按向右键,则打印
RIGHT 按左上键,则打印
UP LEFT
……
```

4.9.2 操纵器功能控制指令(STRIG)

操纵器上除方向键外还有六个功能键,分别为"开始暂停键(START)"、"选择键(SELECT)"、"单发键(A、B)"、"连发键(A、B)"。它们分别由STRIG函数给出键值。

格式:STRIG(X)

类型:函数

简写:STRI.

功用:控制操纵器的"START、SELECT、A、B"六个功能键。

注释:STRIG取自STRIGGER,为"触发"的意思。

X为选定操纵器的代码,取值0、1。分别表示Ⅰ、Ⅱ号操纵器。X可以取数值、变量或表达式。

功能键对应的键值如下表:

表 4—11　功能键、键值对应表

键　　名	键　值
开始暂停键(START)	1
选择键(SELECT)	2
单、连发键(A)	8
单、连发键(B)	4

示例

```
10 REM "STRIG"
20 CLS
30 T=STRIG(0)
40 IF T=1 THEN PRINT "SART"
50 IF T=2 THEN PRINT "SELECT"
60 IF T=4 THEN PRINT "B"
70 IF T=8 THEN PRINT "A"
80 GOTO 30
90 END
RUN ↙
```

每按一个功能键，则打印出该键的名称。

例 4.22　用 I 号操纵器控制玛莉哥哥运动。

```
10 REM "No. 4--26"
20 CLS:SPRITE ON:CGSET RND(2),RND(3)
30 T=STRIG(0):IF T<>1 THEN 30
40 X=35:Y=130:I=1:GOSUB 200
50 K=STICK(0):IF K=0 GOTO 50
60 IF K=1 THEN I=3
70 IF K=2 THEN I=7
80 IF K=4 THEN I=5
90 IF K=5 THEN I=4
100 IF K=6 THEN I=6
110 IF K=8 THEN I=1
```

```
120 IF K=9 THEN I=2
130 IF K=10 THEN I=8
140 X=XPOS(0):Y=YPOS(0)
150 GOSUB 200
160 GOTO 50
170 END
200 DEF MOVE(0)=SPRITE(0,I,1,4,0,0)
210 POSITION 0,X,Y
220 MOVE 0
230 IF MOVE(0)=-1 THEN 230
240 RETURN
RUN ↙
```

程序运行后，必须先按“START”键才能向下运行，这时可使用方向键控制玛莉向八个方向运动。

程序 No. 4—26 中若再加入开始提示，效果将更好。如加入 35 句

```
35 LOCATE 8,10:PRINT "START!!!":
PAUSE 100:CLS
```

则按开始键后，打印“开始”提示，清屏，显示 SP。

STICK、STRIG 在游戏程序中是极有用的指令，我们将在游戏程序编程技巧中详细介绍。

4.10 调入游戏指令(GAME)

在我们的学习机中，已固化了四个不完全的游戏程序，程序名分别为“0”、“1”、“2,1”、“3”。进入直调 BACIC 状态后，系统自动把运行这四个程序定义给 F1～F4 四个功能键，我们只要按其中的任一个，就能把与该功能键对应的游戏程序调入 RAM 中(可编程随机存储器)，并使之运行。这种调入程序的方法，操作简便(只须

按一个键），调入速度快，且调入后自动运行，使丝毫不懂计算机的朋友也可以操作。但也存在较大的缺点：其一，我们的学习机一共只有八个自定义功能键，被四个游戏占去了一半，只剩下四个供用户定义使用，从而使自定义功能键的作用只发挥了50%。其二，在编程中，需要系统复位时，操作稍不小心，就把第二个游戏程序调入了用户内存，从而冲掉了已辛苦编写的程序。（由于在系统复位时是同时按 Ctrl＋Alt＋F2，若 F2 键按得稍早一点，就产生上述恶果。）另外在输入操作中，使用功能键稍不留神，也会调入游戏程序。所以，在编程前，一般要把 F1～F4 四个功能键重新定义。这样做一方面便于输入程序，另一方面也避免了误操作破坏程序。

更改了功能键定义后，方便了编程，但也出现了另外的问题。当我们想调入游戏程序时则又要把功能键重新定义一番。另外，当系统固化有更多的游戏程序时，例如 10 个，则要反复定义功能键，才能调出所有的程序。有没有更简便的调程方法呢？GAME 指令解决了这一难题。GAME 就是专为调入游戏程序而设的特殊指令。

格式：GAME〈表达式〉

类型：命令

简写：GA.

功用：调进游戏程序并运行之。

GAME 命令可把只读存储器（EPROM）中的程序调入用户程序存储器（RAM）中。

注释：GAME 是游戏的意思。

表达式是游戏程序的名称，它可以是数字或字符串。

示例：

GAME 0 ↙　　则调入第一个游戏程序。

GAME 3 ↙　　则调入第四个游戏程序。

GAME 2.1 ↙　　则调入第三个游戏程序。

4.11 综合示例

结合本章介绍的指令，特列举一个游戏程序片断。

片断内容模仿采蘑菇游戏的画面和过程。用 I 号操纵器控制玛莉左、右运动和跳跃。画面上有三个乌龟自右向左运动，碰到玛丽则消失，并在屏幕右上角打印得分。若玛莉跳到屏幕外边，则自动重新启动开始。为便于阅读和理解，程序的有关部分进行了中文注释。

注：由于我们现在掌握的指令尚少，示例仅是模仿，不构成游戏。

```
0 REM "EXAMPLE"
2 CLS:LOCATE 8,10:PRINT "Press START"
4 K=STRIG(0):IF K<>1 THEN 4
6 CLS:LOCATE 8,10:PRINT "START!!!":PAUSE 300:LOCATE2,20
8 INPUT "INPUT 1—BGTOOL,2—VIEW";A
10 ON A GOTO 12,14
12 BGTOOL
14 CLS:SPRITE ON:CGEN(2):CGSET 0,1:X2=56:Y2=150:Q=0
20 VIEW
30 FOR I=4 TO 6
40 DEF MOVE(I)=SPRITE(13,7,3,255,0,I-3)
50 POSITION I,145+I*15,150
60 MOVE I:NEXT
(30～60 行定义三个乌龟从右向左爬行。)
70 K=STRIG(0):G=STICK(0) : IF K=0 AND G=0 GOTO 70 : (读 I
   操纵器)
80 IF K=8 THEN 380 (若按 A 键，转跳子程序)
90 IF G=1 THEN S=3
100 IF G=2 THEN S=7 (90、100 行定义左、右运动)
```

```
110 X1=X2:Y1=Y2
    (引入 X1、X2、Y1、Y2 便于动作前后比较座标,以决定左、右跳。)
120 DEF MOVE(0)=SPRITE(0,S,1,2,0,0)
130 POSITION 0,X2,Y2
140 MOVE 0
    (定义玛丽按操纵器方向运动。)
150 IF MOVE(0)=-1 THEN 150
160 GOSUB 440 (转卡通相遇处理)
170 X2=XPOS(0):Y2=YPOS(0):GOTO 70
180 I=0
190 I=I+1:X2=X2+I/2:Y2=Y2-I*2:IF I>7 THEN 230
200 GOSUB 410 (转出界处理)
210 DEF SPRITE 1,(2,1,0,1,0)=CHR$(13)+CHR$(12)+CHR
    $(15)+CHR$(14)
220 SPRITE 1,X2,Y2:PAUSE 3:SPRITE 1:GOTO 190
(180~220 行定义玛丽向右跳上去)
230 PAUSE 6:I=0
240 I=I+1:X2=X2+I/2:Y2=Y2+I*2:IF I>7 THEN 270
250 GOSUB 410
260 SPRITE 1,X2,Y2:PAUSE 3:SPRITE 1:GOTO 240
270 PAUSE 6:SPRITE 1:POSITION 0,X2,Y2:MOVE 0:GOTO 70
(230~270 行为玛丽向右跳下)
(180~270 行为玛丽面向右跳一次)
280 I=0
290 I=I+1:X2=X2-I/2:Y2=Y2-I*2:IF I>7 THEN 330
300 GOTO 40
310 DEF SPRITE 2,(1,1,0,0,0)=CHR$(12)+CHR$(13)+CHR
    $(14)+CHR$(15)
320 SPRITE 2,X2,Y2:PAUSE 3:SPRITE 2:GOTO 290
330 PAUSE 6:I=0
340 I=I+1:X2=X2-I/2:Y2=Y2+I*2:IF I>7 THEN 370
```

```
350 GOSUB 410
360 SPRITE 2,X2,Y2:PAUSE 3:SPRITE 2:GOTO 340
370 PAUSE 6:SPRITE 2:POSITION 0,X2,Y2:MOVE 0:GOTO 70
(280～370 为面向左跳一次子程序。)
380 ERA 0 (令 0#动作消失)
390 IF X1<X2 THEN 180 (转面向右跳)
400 IF X1>X2 THEN 280 (转面向左跳)
(380～400 为左、右跳判断处理子程序)
410 IF X2<16 OR X2>240 THEN X2=100:GOTO 14
420 IF Y2<24 OR Y2>220 THEN Y2=150:GOTO 14
430 RETURN
(410～430 判断 0#动作是否出界及出界处理。)
440 IF CRASH(0)=4 THEN CAN 4:Q=Q+1
450 IF CRASH(0)=5 THEN CAN 5:Q=Q+1
460 IF CRASH(0)=6 THEN CAN 6:Q=Q+1
470 LOCATE 12,0:PRINT "SCORE:";
480 LOCATE 20,0:PRINT Q
490 IF MOVE(4)=0 AND MOVE(5)=0 AND MOVE(6)=0 THEN 30
500 RETURN
(440～460 是判断 SP 是否相遇及相遇后的处理程序)
(470～480 是打印得分程序)
(490～500 测试若 4、5、6 号动作完毕则重新启动)
```

程序输入完毕，检查无误后，可运行程序。

```
RUN ↙ 显示
Press START
```

按“START”键后，清屏，打印

```
START!!!
```

延时一段时间后，在屏幕底部打印

```
INPUT 1—BGTOOL,2—VIEW? □
```

我们键入 1 则进入 BG GRAPHIC 状态。按键控绘图方法依照

图 4—13 绘出如图 4—14 的背景。

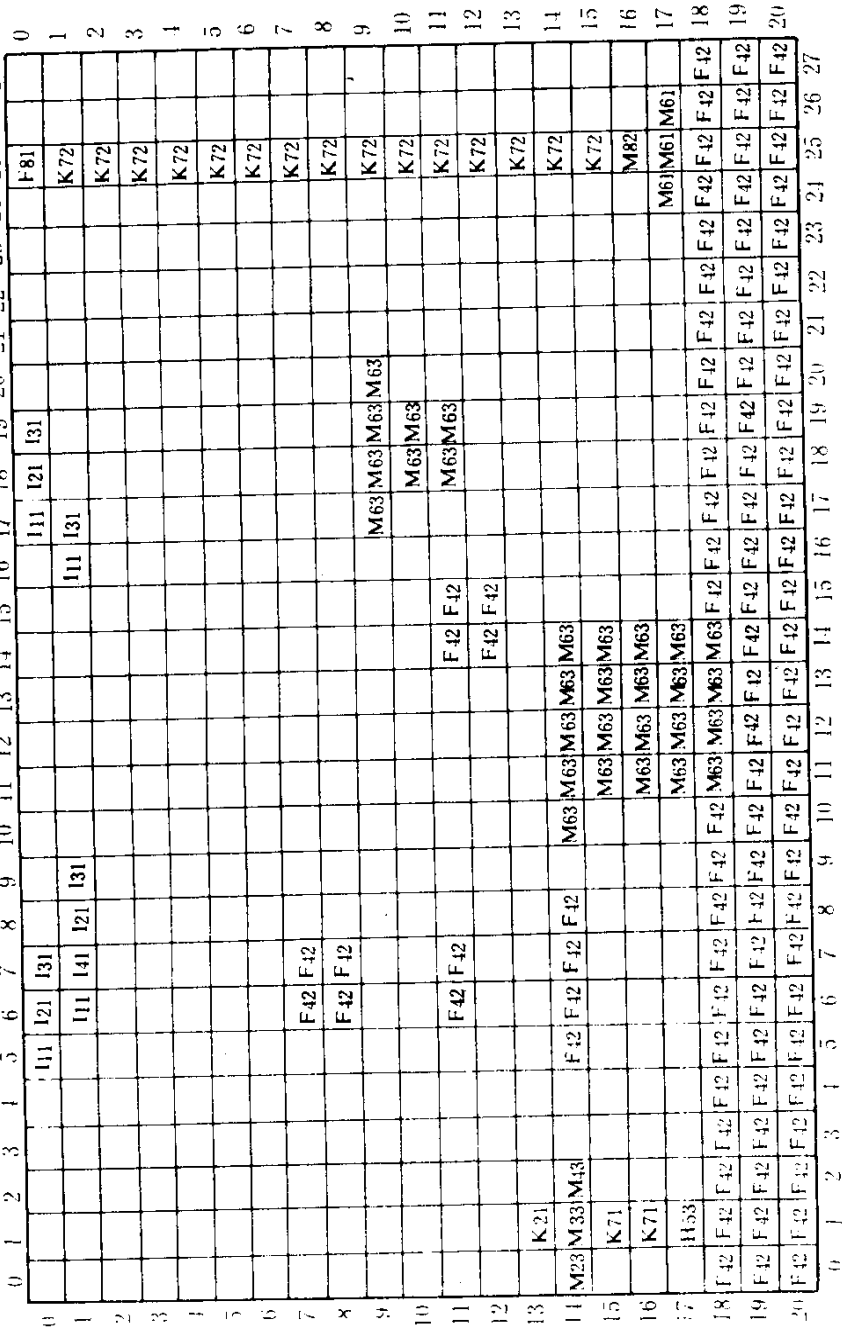


图 4—13　绘图规则

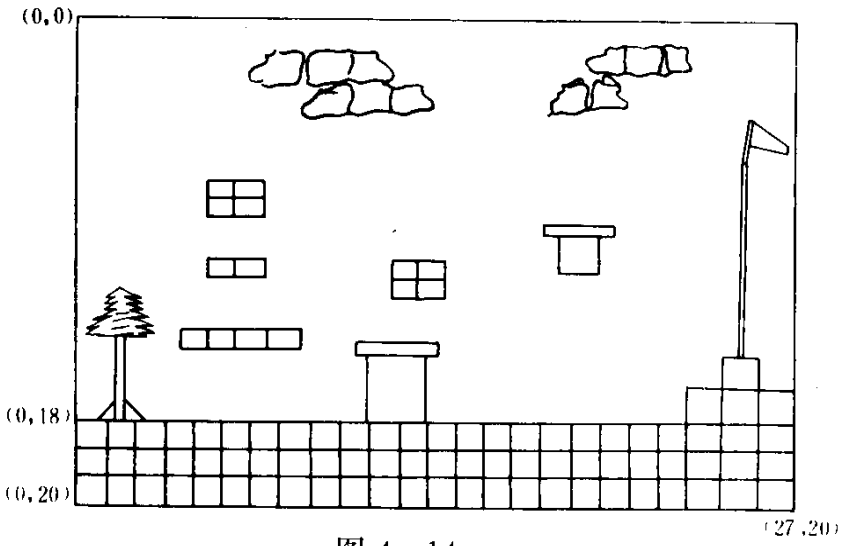


图 4—14

绘图完毕，按 Esc 键、Stop 键，返回 F BASIC 状态。重新运行程序，待出现

INPUT 1—BGTOOL,2—VIEW

时，键入 2，则清屏，调出已绘背景，有三个乌龟从右向左运动。当按动操纵器的方向键或 A 键时，出现玛莉随操纵器的控制而运动。当玛莉与乌龟相遇时，乌龟消失，并在屏幕右上方打印得分“SCORE＝1”……。当按 A 键可令玛莉跳起来。当玛莉运动到边界时若继续前进则程序又从 14 行开始……。

本游戏片断虽没有构成游戏，但可使读者一窥游戏程序的奥秘，对游戏程序的编写增加初步的认识。

4.12 本章要点说明

在本章中，我们学习了 38 条指令，这些指令体现了 F BASIC 语言与其它 BASIC 语言的根本区别，它们是 F BASIC 语言的中心内容，同时也是 F BASIC 语言的重点和难点。为便于读者理解、掌握，特把本章的三点重要概念综述如下：

4.12.1 显示卡通与动作卡通的区别

我们知道在SP面(高分辨图形面)上共有两种卡通,即显示卡通和动作卡通。这两种卡通分属两个不同的范畴,它们各有专用的控制指令。

1. 显示卡通

显示卡通由DEF SPRITE语句定义,由SPRITE语句决定显示座标并启动显示,由SPRITE n语句解除定义。

2. 动作卡通

动作卡通由DEF MOVE语句定义,由POSITION语句决定动作起点座标,由MOVE指令启动运动,由CAN指令解除定义。同时动作卡通还受CUT指令作用而停止运动,受ERA指令作用而停止运动和消失。

3. 两种卡通都必须用SPRITE ON指令打开显示开关,用SPRITE OFF指令关闭开关。

4. 由于显示SP和动作SP分属不同范畴,所以在同一SP面最多可显八个显示卡通、八个动作卡通。

例4.23 编一程序在同一画面中显示出16类卡通。

```
10 REM "No. 4—28"
20 CLS:SPRITE ON
30 SCREEN 1,0:Y=-1
40 FOR I=148 TO 204 STEP 8
50 Y=Y+1
60 DEF SPRITE Y,(Y/2,1,0,0,0)=CHR$(I)+CHR$(I+1)+CHR
   $(I+2)+CHR$(I+3)
70 SPRITE Y,50*(1-Y*(Y<4)-(Y-4)*(Y>3)),50*(1+Y/4)
80 NEXT
90 FOR I=0 TO 7
100 DEF MOVE(I)=SPRITE(I,I+1,1,1,0,I/2)
110 POSITION I,50*(1-I*(I<4)-(I-4)*(I>3)),I/4*50+150
```

```
120 MOVE I:NEXT
RUN ↙
```

每行显示四个 SP，共四行显示 16 个 SP。

4.12.2 配色的实现

在我们已学习的语句中，与配色有关的共六条，分别为：PALETB、PALETS、CGSET、COLOR、DEF SPRITE、DEF MOVE。它们之间的关系及使用搭配方法如下：

1. 若系统原定义的 BG、SP 各调色板的颜色组合满足使用需要，则转以后的工作；若不符合使用要求，则使用 PALETB 重新设定 BG 调色板的配色号码，使用 PALETS 设定 SP 的配色号码。

2. 由 CGSET 语句分别选定 BG、SP 的调色板，以供 COLOR、PALETB、PALETS、DEF SPRITE、DEF MOVE 各语句使用。

3. 若欲对背景图案的某一部分染色，只能使用 COLOR 指令。COLOR 语句仅能对背景图案染色，而不能对底背景染色。

4. 关于显示卡通的色彩，只能由 DEF SPRITE 语句选定。

5. 关于动作卡通的色彩，则只能通过 DEF MOVE 语句选取。

6. 底背景在复位后(即开机后的状态)为黑色。当需要改变底背景颜色时，可使用 PALETB 语句从 52 种颜色表中任选一种。定义底背景颜色的格式为：

```
PALETB 0,XX,XX,XX,XX
```

其中第一个“XX”决定底背景颜色，后三个“XX”决定 BG 面由 CGSET 语句选定调色板中的 0 号配色号码的颜色组合。

4.12.3 各显示面的正确使用

F BASIC 系统的显示面是较为独特的，它共有四层，见图 4 15：

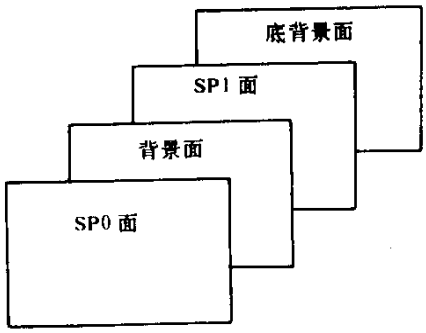


图 4—15　F BASIC 显示面示意图

• 第一层为卡通 0 面，用 SPRITE 或 DEF MOVE 指令可把 SP 调到这个面上来。SP 面的大小按点计算：X 方向为 240 点，Y 方向为 220 点（指显示有效范围，实际范围为 256X256 点）。另外，当 SP 为 1 个字符大小时可取 228 点。

• 第二层为背景面，BASIC 的指令和程序就输入这一层显示，BG GRAPHIC 绘图也在这一层。这一层有两个显示页面，可通过 SCREEN 指令把任一页面调到前面来（注：BS. 2A 版本不能调出）。背景面大小按字符计算，X 方向为 28 个字符，Y 方向为 24 个字符。

• 第三层为 SP1 面。游戏中，SP 要隐藏到背景之后，这就要使 SP 在 SP1 面显示，SP1 面显示的卡通可以被背景遮住。SP1 面的大小与第一层相同，这两层统一称为 SP 面。

• 第四层为底背景面。这一层适合表示天空、陆地和大海的效果。整个显示面用一种颜色表示。开机后，系统自动将其设定为黑色，可使用 PALETB 指令将其设置为任一色。底背景的大小按字符计算为：X 方向 32 个字符，Y 方向 30 行。按点计算则为：X＝256 点，Y＝240 点。

SP 面、背景面、底背景面的对应关系见图 4—16。图中，中间的

矩形框为背景面，其四个角的座标为：左上(0,0)、右上(27,0)、左下(0,23)、右下(27,23)。阴影块为SP面，其四个角的座标为：左上(0,0)、右上(240,0)、左下(0,220)或(0,228)、右下(240,220)或(240,228)。最外边的矩形框为底背景面，其四个角的座标为：左上(0,0)、右上(32,0)或(256,0)(按点计)、左下(0,30)或(0,240)、右下(32,30)或(256,240)。

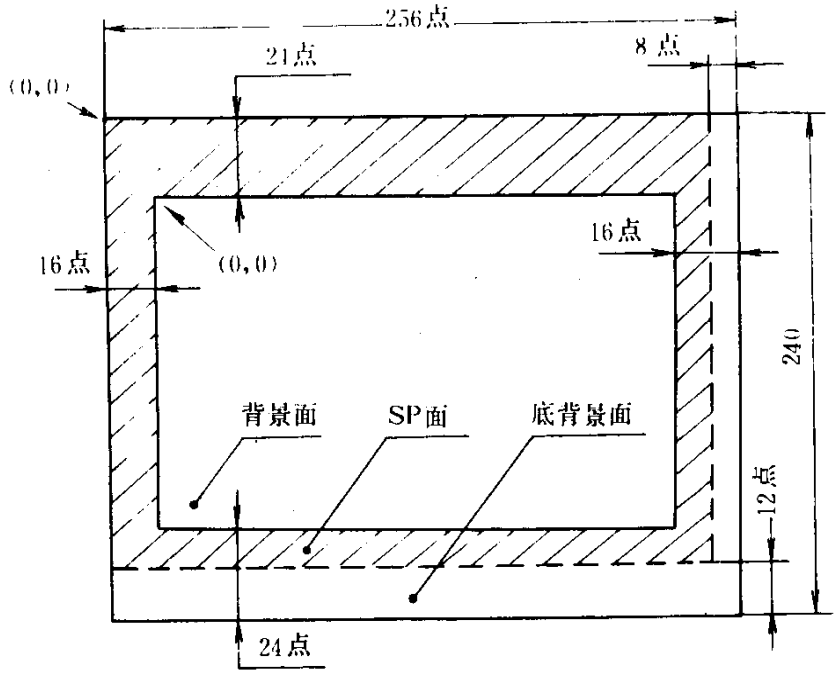


图 4—16　背景、底背景、SP三面的对应关系

下面重点介绍背景面、SP面的使用方法。

1. 背景面(BG面)

一般情况下，在BG面上显示的是各种背景图案，但是卡通图案也可以使用BG面。当SP使用BG面时，各SP将失去其原有的形象，而以其符号代码所对应于符号表B中的字符或图案显示。也就是说，SP在BG面显示时，显示的是字符或背景图案。见例4.23

例 4.24　在BG上显示SP。

```
10 REM "No. 4—29 SP→BG"
20 CLS:SPRITE ON:CGEN3
30 FOR I=0 TO 7
```

```
40 DEF SPRITE I,(I/2,1,0,0,0)=CHR$(I*4)+CHR$(I*4+1)+
   CHR$(I*4+2)+CHR$(I*4+3)
50 SPRITE I,I*20+50,I*8+50
60 NEXT
RUN ↙
```

画面上只显示 32 个背景图案。

当使用动作 SP 定义语句，用 MOVE 指令启动 SP 运动时，这时运动的已不再是卡通，而是四个字符或背景图案。如例 4.24。

例 4.25 定义动作卡通在 BG 面运动。

```
10 REM "No. 4—30"
20 CLS:SPRITE ON:CGEN 3
30 DEF MOVE(0)=SPRITE(3,7,1,255,0,0)
40 MOVE 0
RUN ↙
```

可以看到，这时运动的已不再是 3 号 SP 飞鱼，而是"DEFG"四个字母在运动。

2. SP 面

SP 面由 SPRITE ON 指令打开，可显示两类 SP，即显示卡通和动作卡通。

字符和背景图案也可以使用 SP 面。同 SP 使用 BG 面一样，当字符或背景使用 SP 面时，字符也将失去它原来的定义，而以 SP 的一部分显示在 SP 面上，见例 4.25。

例 4.26 使字符显示在 SP 面上。

```
10 REM "No. 4—31"
20 CLS:CGEN 0
30 FOR I=0 TO 672
40 PRINT CHR$(225);:NEXT
50 LOCATE 0,0:PRINT"D"
60 END
```

RUN ↙

屏幕打印一黑色方块，在方块中显示了飞鱼的头部。可见，这时的字母已失去本来面目。

通过例 4.25 程序的运行，我们找到了不使用显示 SP 的定义语句 DEF SPRITE 来显示 SP 的方法，即直接使用输出语句 PRINT 显示 SP。

例 4.27 用输出语句显示 SP。

```
10 REM "No. 4—32"
20 CLS:CGEN 0
30 FOR I=0 TO 672:PRINT CHR$(225);:NEXT
40 LOCATE 10,10:PRINT "DE"
50 LOCATE 10,11:PRINT "FG"
60 END
```

RUN ↙

在屏幕显示出飞鱼的图形。

例 4.26 给我们一个启示，使我们找到了绘制新卡通图案的方法。

你见过一头四身的乌龟吗？见过从身体中间爆炸的人吗？运行 No. 4—33，便可看到这一奇观。

例 4.28 自行设计新的卡通图案。

```
10 REM "No. 4—33 Design SP"
20 CGEN 0:CLS
30 FOR I=0 TO 672
40 PRINT CHR$(225);:NEXT
50 LOCATE 10,10:PRINT "45"
60 LOCATE 10,11:PRINT CHR$(180);CHR$(181)
70 LOCATE 10,12:PRINT CHR$(182);CHR$(183)
80 LOCATE 10,13:PRINT "67"
90 LOCATE 10,4:PRINT CHR$(184);CHR$(185);CHR$(189);CHR
   $(185);CHR$(189)
```

```
100 LOCATE 10,5:PRINT CHR$(186);CHR$(187);CHR$(191);CHR
    $(187);CHR$(191)
110 END
RUN ↙
```

则显示程序开头描述的景象。相信读者使用这种方法，定会拚凑组合出更有趣的动画。

通过以上的示例可见，决定 SP 显示于 BG 面和字符显示于 SP 面的关键是 CGEN 指令。也就是说，必须使用 CGEN 指令才能实现 SP 显示于 BG 面，字符显示于 SP 面。

以上小结，但愿能对读者加深理解 F BASIC 语言有所帮助。

第五章　F BASIC 语言的深入和提高

前四章介绍了 F BASIC 语言的 74 条指令，其中有 10 条是未公布的指令，有 24 条指令是 F BASIC 语言所独有的。使用这 74 条指令可以编制一些较复杂的运算程序和简单的游戏程序。但是，当进行较复杂的程序设计、尤其是游戏程序设计时，仅使用这 74 条指令，往往不能达到预期目的。如：使卡通在规定的图形中运动；加配音乐使游戏更加生动逼真等等都难以实现。可以这样说，前四章为 F BASIC 语言的入门篇——基础部分，第五章为 F BASIC 语言的深入篇——提高部分。本章将重点介绍字符处理、出错处理、音响处理等指令，并补充介绍完 F BASIC 语言的其它指令。在弄通本章介绍的内容后，可以说你已经基本掌握了 F BASIC 语言的真谛。那么，再编制程序时，就会随心所欲，水到渠成了。

5.1　键盘扫描指令(INKEY $)

格式：INKEY $ (n)

类型：函数

简写：INK.

功用：对键盘进行扫描，等待读取输入的某一字符。如果在该函数的执行瞬间，用户按下了某一键，则该键的字符就会赋给 INKEY $ 函数。这样，程序就可以根据 INKEY $ 取值情况，决定转移方向。

注释：INKEY $ 取自于 Input from keyboard，是从键盘上输入的意思。

INKEY $ 与 INPUT 语句不同，它无须等待用户按 ENTER 键（回车）就可以读取一个字符。如果在 INKEY $ 执行期间，用户没按任何键，则 INKEY $ 读入一个空字符。

当程序执行到 INKEY $ 函数时，用户按下的键值并不在屏幕上显示出来。

n 是一个哑变量，它可以取－32768～＋32767 之间的任意数，也可以是变量或表达式（注：采用变量或表达式无实用意义）。

n 值可以省略。当省略 n 值时，若用户不按键，则函数值为 0，并继续执行 INKEY $ 以后的语句；当使用 n 时，则程序在执行到 INKEY $ 语句时，暂停运行，等待用户从键盘上输入一个字符后才向下运行。

示例：

```
10 REM "INKEY $"
20 FOR I=1 TO 100
30 PRINT I" "
40 A$=INKEY$ : A=ASC(A$)
50 PRINT A$"="A,
60 NEXT
70 END
RUN          ↙
```

若不按任何键，则打印

```
1 =0 2 =0 3 =0
4 =0 ……
```

若我们按"A"键，则打印

```
5 A=65 6=0
```

若我们按"U"键，则打印

```
8 U=85 ……
```

运行结果表明，当 INKEY $ 函数省略 n 时，程序执行到 INKEY $ 语句时，所读的值是 0，并继续以后程序的运行。

若把上述程序 40 行改为：

```
40 A$=INKEY$(0):A=ASC(A$)
```

运行程序后，则显示

```
1□
```

打印 1 后，出现一闪烁光标，等待用户输入。若我们按 A 键，则打印

```
1 A=65 2□
```

打印 A 的键值，又处于等待用户的输入状态。若连续键入字符，则连续打印字符所对应的键值。直到循环 100 次后才停机。

运行结果表明，如果 INKEY $ 函数使用 n，当程序执行到 INKEY $ 时要暂停，等待用户输入，只有读到一个键值后，才执行下一语句。

例 5.1 编一程序，说明 INKEY $ 函数读键功能的应用。

```
10 REM "No. 5-1"
20 CLS
30 A$=INKEY$(0)
40 IF A$="0" THEN PRINT "OK"
50 IF A$="N" THEN PRINT "No"
60 IF A$="E" THEN PRINT "END":END
70 GOTO 30
80 END
RUN↙   屏幕上显示闪烁光标
□      键入 O 则打印
OK     键入 N 则打印
No     键入 A 则没有变化
       键入 E 则打印
END
OK     停机
```

程序 No. 5—1 给我们一点启示，它启发我们找到了不使用操纵器也能够控制 SP 运动的方法。即使用键盘控制 SP 运动。

例 5. 2 编程使用键盘控制 SP 运动。

假定用 U、D、L、R 四个键控制 SP 的上、下、左、右运动。编程如下：

```
10 REM "No. 5—2 Use Keyboard to control sp"
20 CLS:SPRINTE ON
30 SCREEN 0,1
40 A$=INKEY$(0):B=0
50 IF A$="U" THEN B=1
60 IF A$="D" THEN B=5
70 IF A$="R" THEN B=3
80 IF A$="L" THEN B=7
90 X=XPOS(0):Y=YPOS(0)
100 DEF MOVE(0)=SPRITE(0,B,1,4,0,RND(4))
110 POSITION 0,X,Y:MOVE 0
120 GOTO 40
130 END
RUN ↙
```

按 U、D、R、L 四键可控制 SP 分别向上、下、右、左四个方向运动。从而实现了使用键盘操纵 SP 的愿望。

通过操作，我们发现使用 U、D、L、R 四键很不方便。学习机的键盘有专用于控制光标移动的四个键，即"→←↓↑"四键。能否用这四个键控制 SP 运动呢？回答是肯定的。但在使用前，首先要找出这四键的符号代码。

例 5. 3 编程测试键盘代码。

使用 INKEY$ 函数可方便地测出各键的代码。编程如下：

```
10 REM "No. 5—3 Check Key Value"
20 A$=INKEY$(0):A=ASC(A$)
```

```
30 PRINT A$"="A,
40 GOTO A$
50 END
RUN ↙  出现一闪烁光标。
```

我们按向上的箭头，则打印

=30　　我们按向下的箭头，则打印

=31

从而可很快测出向左的键值=29，向右的键值=28。

用同样方法，可测出键盘上各键对应的代码。为便于使用，特将各键代码列于表5—1。由于各符号的代码，诸如A、B、C、+、—、*等已于符号代码表A中列出，故略去，仅列出其它键的代码。

了解了键的代码，就可以用光标控制键控制SP动作了。

表5—1　按键——ASCII码对应表

代码(10进制)	代码(16进制)	对应的按键	代码(10进制)	代码(16进制)	对应的按键
0	00		17	11	Ctrl+Q
1	01	STOP	18	12	Ctrl+R,Ins
2	02	Ctrl+B	19	13	Ctrl+S
3	03	Ctrl+C	20	14	Ctrl+T
4	04		21	15	Ctrl+U
5	05	Ctrl+E	22	16	Ctrl+V
6	06	Ctrl+F	23	17	Ctrl+W
7	07	Ctrl+G	24	18	Ctrl+X
8	08	Del	25	19	Ctrl+Y
9	09	Ctrl+I,Tab	26	1A	Ctrl+Z
10	0A		27	1B	Esc
11	0B	Home	28	1C	→
12	0C	Ctrl+L,Home+Shift	29	1D	←
13	0D	Ctrl+M+Enter	30	1E	↑
14	0E	Ctrl+N	31	1F	↓
15	0F	Ctrl+O	32	20	空格(Space)
16	10	Ctrl+P			

例 5.4 用方向键控制 SP 运动。

```
10 REM "No. 5—4,Control SP's Movement"
20 CLS : SPRITE ON : SCREEN 0,1
30 A$=INKEY$(0) : A=ASC(A$) : B=0
40 IF A=30 THEN B=1
50 IF A=31 THEN B=5
60 IF A=29 THEN B=7
70 IF A=28 THEN B=3
80 X=XPOS(0) : Y=YPOS(0)
90 DEF MOVE(0)=SPRITE(0,B,1,4,0,RND(4))
100 POSITION 0,X,Y : MOVE 0
110 GOTO 40
120 END
RUN ↙
```

现在可以用四个方向键控制 SP 运动了。

5.2 字符处理指令

F BASIC 语言的字符处理指令共有七条，分别为串型数据转换为数值的函数(VAL)；将数值转换为相应的字符串的函数(STR$)；测量字符串长度的函数(LEN)；从字符串左边取子串的函数(LEFT$)；从字符串右边取子串的函数(RIGHT$)；从字符串的任一位置取子串的函数(MID$)；和字符串检索函数(INSTR)。

字符处理函数是本章的重点内容，它们在绘图中，在数学的高精度运算程序中将发挥重要的作用。

5.2.1 串→数转换函数(VAL)

格式：VAL(字符串)

类型：函数

简写：VA.

功用：

1. 将十六进制数转换成相应的十进制数。

2. 将字符串型的数据转换成十进制数。就是说，VAL 函数具有去掉字符串引号、字符的功能。VAL 函数返回的值可以参加数值运算。

注释：VAL 源于 Value，是数值的意思。

1. 当 VAL 函数用于进制转换时，16 进制数必须在数值前面加“&H”，且用双引号引起来。如

“&H32”

“&HAF00”

等。16 进制数的取值范围为 &H0～&HFFFF。当取值为 &H0～&H7FFF 时，返回的十进制数是 0～＋32767；当取值为 &H8000～&HFFFF 时，返回的十进制数则是－32768～－1。

示例：

把 &HFF 转换为 10 进制数，则有

```
10 A=VAL("&HFF")
20 PRINT A
RUN        ↙      打印
255
```

或者

```
10 PRINT VAL("&HFF")
```

2. 当 VAL 函数用于串型数据转换时，其返回值仅保留字符串前面的数码部分，字符部分则删去。

下面给出串变量 A$ 取不同值的情形下，VAL 函数返回的结果。

A$	VAL(A$)
“1235”	1235
“－367”	－367
“45CED”	45
“HELLO”	0

例 5.5 编程演示 VAL 函数的串→数转换功能。

```
10 REM "No. 5—5,character to Data"
20 INPUT "INPUT CHARACTER:";A$
30 A=VAL(A$)
40 PRINT "A$ ="A$ ,"VAL("A$ ")="A,
50 GOTO 20
60 END
RUN    ↙    显示
INPUT CHARACTER:? □
键入 34ABC    ↙    打印
A$ =34ABC VAL(34ABC)=34
INPUT CHARACTER:? □
键入 7AB38    ↙    打印
A$ =7AB38 VAL(7AB38)=7
INPUT CHARACTER:? □
键入 GOOD    ↙
A$ =GOOD VAL(GOOD)=0
…… 键入 AB3214    ↙
A$ =AB3214 VAL(AB3214)=0
……
```

可见，VAL 函数就是保留字符串开头的十进制数，后面的字母、符号一概删去；若第一个字符不是数字，即使后面全是数字，VAL 函数的返回值仍然是 0。

为了说明 VAL 函数的返回值是数字，再看一例。

例 5.6 VAL 函数返回值相加演示。

```
10 REM "No. 5—6,VAL"
20 INPUT "INPUT A$ ,B$ :";A$ ,B$
30 A=VAL(A$ ) : B=VAL(B$ )
40 C=A+B : C$ =A$ +B$
50 PRINT A$ "+"B$ "="C$ " CHAR. ",A"+"B"="C"DATA",
```

```
60 GOTO 20
70 END
RUN    ↙      显示
INPUT A$,B$ : ? □
键入 324B,1253
324B+1253=324B1253 CHAR.
324+1253=1577 DATA
INPUT A$,B$ : ? □
键入 12ABCDE,CDEFG    ↙
12ABCDE+CDEFG=12ABCDECDEFG CHAR.
12+0=12
.....
键入 A12345,B762378

A12345+B762378=A12345B762378
0+0=0
.....
```

通过演示，可清楚地看到 VAL 函数的作用。

5.2.2 数→串转换函数(STR$)

格式：STR$(X)

类型：函数

简写：STR.

功用：将十进制数转换成相应的字符串。

注释：STR$取自于 String，为串的意思。

X 的取值范围为－32768～＋32767。可以取常数、变量或表达式。STR$函数会将 X 的值转换成相应的串形数。

如果 X 为负值，则 STR$生成的字符串的第一个字符为“－”；如果 X 为正值，则 STR$生成的字符串的第一个字符为空格。

例 5.7 编程演示 STR$函数的功能。

```
10 REM "No. 5－6 STR$"
20 CLS
30 INPUT "A,B=";A,B
40 PRINT A"＋"B"＝"A＋B
50 PRINT "STR$("A")＋STR$("B")＝";STR$(A)＋STR$(B)
60 GOTO 30
RUN  ↙      显示
A,B=? □     键入 125,135
125＋135＝260
STR$(125)＋STR$(135)＝125135
A,B=? □     键入 3212,－215
3212－215＝2997
STR$(3212)＋STR$(－215)＝3212－215
……
```

可见，通过 STR$ 函数的处理，原来的十进制数 125、135 虽然外形没有改变，但已发生了质的变化。这时的 125、135 已不再是数字而变为字符了。因此，当它们相加时，不再按数字加法规则而按字符串的规则相加了。

有些资料介绍 STR$ 函数与 VAL 函数相反，是 VAL 函数的反运算。有些键盘随机手册中介绍 STR$ 函数的功用是"将十进制数转换成十六进制数的字符串"。前一种说法不全面，后一种说法是错误的。程序 No. 5－6 说明 STR$ 函数只将十进制数变成相应的字符串，不能变成十六进制数。如十进制数 255 对应的十六进制是 &HFF，而 STR$(255)＝"255"，显然不能转成 16 进制数。前一种说法只对于纯数字的字符串有效。如：

A$＝"1234" ……1234 为字符串

A＝VAL(A$)＝1234 ……1234 为十进制数

STR$(A)＝"1234" ……1234 又变成字符串

这时

A$＝STR$(VAL(A$))＝A$

STR $ 确实是 VAL 函数的逆运算。

但是对数值、字母混合的字符串则无效。如：

B $ ="12BAC" ……12ABC 为混合字串

B=VAL(B $)=12 ……12 为十进制数

STR $ (B)="12" ……仅 12 变为字符串

可见 STR $ 函数已不能令 B 再变为"12ABC"，所以说这时 STR $ 已不是 VAL 函数的逆运算。

5.2.3 测试字符串长度函数(LEN)

格式：LEN(字符串)

类型：函数

简写：LE.

功用：计算出字符串的字符个数。

使用 LEN 函数可极方便地求出字符串长度。

注释：LEN 源于 Length，是长度的意思。

字符串也可以是变量或表达式。当直接测试字符串的长度时，整个字符串必须用双引号括起来。如

LEN("ABCD+－/＊FG")

LEN 函数把字符串内的符号、空格均作为字符对待。测量字符串的最大长度为 31。

当字符串以表达式的形式测试时，如

LEN(A $ +B $)

A $ 和 B $ 的字符长度之和也不能大于 31。

示例：

```
10 REM "LEN"
20 CLS
30 INPUT "PLEASE INPUT A $ : ";A $
40 PRINT A $
50 PRINT "LEN(A $ )=" LEN(A $ )
60 PRINT "LEN(A $ +A $ )=" LEN(A $ +A $ )
```

```
70 PRINT "LEN(A$+A$+A$)=" LEN(A$+A$+A$)
80 PRINT : GOTO 30
90 END
RUN      ↙      显示
PLEASE INPUT A$ : ? □
键入 ABCDEF      ↙则打印
A$=ABCDEF
LEN(A$)=6
LEN(A$+A$)=12
LEN(A$+A$+A$)=18
PLEASE INPUT A$=? □
键入 AB      CD      ↙则打印
A$=AB      CD
LEN(A$)=8
LEN(A$+A$)=16
LEN(A$+A$+A$)=24
PLEASE INPUT A$=? □
键入 AA      BB      CC      DD      ↙则打印
A$=AA      BB      CC      DD
LEN(A$)=13
LEN(A$+A$)=26
LEN (A$+A$+A$)=
?,OV ERROR IN 60 表示 60 行发生了溢出。
```

可见,测试的字符长度不能大于 31。

5.2.4 从字符串左边取子串函数(LEFT$)

格式:LEFT$(字符串,n)

类型:函数

简写:LEF.

功用:从字符串左边取出 n 个字符形成一个子字符串。

注释:LEFT$ 是从左边取字符的意思。

n 为从字符串的左边取字符的个数，n 的取值范围为 0～31，n 可以取数值常数、可以取变量、也可以是表达式。

字符串的字符个数不能大于 31 个。

示例 1：

```
LEFT$("XYZABCD",3)
LEFT$(A$,7)
```

示例 2：

```
10 REM "LEFT$"
20 A$="ABCDEFGHIJK"
30 FOR I=1 TO 11
40 PRINT LEFT$(A$,I),
50 NEXT : END
RUN        ↙        打印
A            AB            ABC            ABCD
ABCEDF       ABCDEF        ABCDEFG        ABCDEFGH
ABCDEFGHI    ABCDEFGHIJ
ABCDEFGHIJK
```

例 5.8 编程打印一字符三角形

```
10 REM "No. 5-7,Triangle"
20 CLS
30 A$="ABCDEFGHIJK"
40 FOR I=1 TO 11 STEP 2
50 A=A+1
60 LOCATE 13-A,5+A
70 PRINT LEFT$(A$,I)
80 NEXT
90 LOCATE 0,20 : END
RUN      ↙      打印
```

A

ABC

ABCDE

ABCDEFG

ABCDEFGHI

ABCDEFGHIJK

5.2.5 从字符串右边取子串函数(RIGHT $)

格式:RIGHT $(字符串,n)

类型:函数

简写:RI.

功用:从字符串右边取出 n 个字符形成一个子字符串。

注释:RIGHT $ 是从右边取字符的意思。

n 为从字符串的右边取字符的个数,n 的取值范围为 0～31,n 可以取数值常数、可以取变量、也可以是表达式。

字符串的字符个数不能大于 31 个。

示例

```
10 REM "RIGHT $"
20 A $ ="ABCDEFGH"
30 FOR I=1 TO 8
40 PRINT RIGHT $ (A $ ,I)
50 NEXT : END
RUN      ↙      打印
H          GH         FGH         EFGH
DEFGH    CDEFGH  BCDEFGH  ABCDEFGH
```

例 5.9 编程打印一字母菱型。

```
10 REM "No. 5－8,Lozenge"
20 CLS
30 A $ ="ABCDEFGHIJK"
40 FOR I=1 TO 11 STEP 2
50 A=A+1
```

```
60 LOCATE 13-A,5+A
70 PRINT LEFT$(A$,I)
80 NEXT
90 FOR I=9 TO 1 STEP -2
100 A=A-1:B=B+1
110 LOCATE 13-A,11+B
120 PRINT RIGHT$(A$,I)
130 NEXT
140 LOCATE 0,20:END
RUN  ↙  打印
```

A

A B C

A B C D E

A B C D E F G

A B C D E F G H I

A B C D E F G H I J K

C D E F G H I J K

E F G H I J K

G H I J K

I J K

K

5.2.6 从字符串中任一位置取子串函数(MID$)

格式：MID$(字符串,m,n)

类型：函数

简写：MI.

功用：从某个较长的字符串中从指定位置抽取指定个数的字符以形成一个子字符串。

注释：MID$是从中间取字符的意思。

m 表示取字符的位置，要从字符串的左边向右数。m 的取值范围为 0～31，它可以是数值常数、可以是变量、也可以是表达式。

n 为从字符串的第 m 位向右取的字符个数，n 的取值范围为 0～31，它可以取数值常数，可以取变量，也可以是表达式。

字符串的字符个数不能大于 31 个。

示例：

```
10 REM "MID$"
20 A$="FCS-90F BASIC"
30 FOR I=1 TO LEN(A$) STEP 2
40 A=A+1
50 LOCATE 5+A*S,8+A
60 PRINT MID$(A$,I,2)
70 NEXT
80 END
RUN     ↙      打印
FC
  S-
    90
      F
        BA
          SI
            C
```

例 5.10 编程画一三角形。

```
10 REM "No. 5-9"
20 A$="FCS-90F-BASIC":CLS
30 FOR I=1 TO LEN(A$) STEP 2
40 A=A+1:B$=MID$(A$,I,2)
50 LOCATE 5+A*2,8+A:PRINT B$
60 LOCATE 5+A*2,10-A:PRINT B$
70 LOCATE 18,5+A:PRINT MID$(A$,I,2)
80 NEXT
90 END
RUN     ↙      打印
```

```
          C
        SI
      BA
    F—    FC
  90      S—
S—        90
FC        F—
  S—      BA
    90    SI
      F—  C
        BA
          SI
            C
```

从示例的结果可以看出，MID $ 语句兼备了 LEFT $ 、RIGHT $ 两函数的功能，且比这两个函数使用更灵活、更方便。

5.2.7 字符串检索函数(INSTR)

格式：INSTR(<被查串>,<子串>)

类型：函数

简写：INS.

功用：检索某一子字符串在另一字符串中的位置。

注释：INSTR 源于英文 Inside String，在这里表示子串在被查串中的位置的意思。

INSTR 检查的结果为一整数值，它代表子串首字符在被查串中的位置。位置座标遵循从左向右数的原则。

被查串、子串的字符个数均不能大于 31 个。它们可以是具体的字符串，可以是串变量，也可是表达式。如：

```
INSTR("ABCDEFG","DE")
INSTR(A$,B$)
INSTR(A$+B$,C$+D$)
INSTR(A$+"BCD",B$)
```

都是合法的。

示例：

```
10 REM "INSTR"
```

```
20 CLS
30 A$="ABCDEABC":B$="BCDEA":C$="DE"
40 AB=INSTR(A$,B$):AC=INSTR(A$,C$):BC=INSTR(B$,C
   $)
50 PRINT "A$=" A$
60 PRINT "B$=" B$
70 PRINT "C$=" C$
80 PRINT "INSTR(A$,B$)="AB
90 PRINT "INSTR(A$,C$)="AC
100 PRINT "INSTR(B$,C$)="BC
110 END
RUN     ↙     打印
A$=ABCDEABC
B$=BCDEA
C$=DE
INSTR(A$,B$)=2
INSTR(A$,C$)=4
INSTR(B$,C$)=3
OK
```

例 5.11　综合示例

设计绘制心形的程序。

```
10 REM "No. 5—10,Heart"
20 CLS:B$="HHBHBBBEBEEE"
30 A$="AIIIIIIIIIIIGIGGGGH"
40 K$="ABCDEFGHI"
50 I$="ABCGHIDEF"
60 X=13:Y=22:I=0
70 GOSUB 100:SWAP A$,B$:GOSUB 100
80 IF I=1 THEN LOCATE 0,0:END
90 IF I=0 THEN SWAP K$,I$:I=1:GOTO 70
100 L=LEN(A$):FOR P=1 TO L
```

```
110 K=P
120 IF I=0 THEN 140
130 K=L+1-P
140 C=INSTR(K$,MID$(A$,K,1))-1
150 V=C/3:H=CMOD 3
160 X=X+H+3*(H=2):Y=Y+V+3*(V=2)
170 LOCATE X,Y:PRINT CHR$(207):NEXT
180 RETURN
190 END
RUN  ↙   打印出一个心形(见图 5—1)。
```

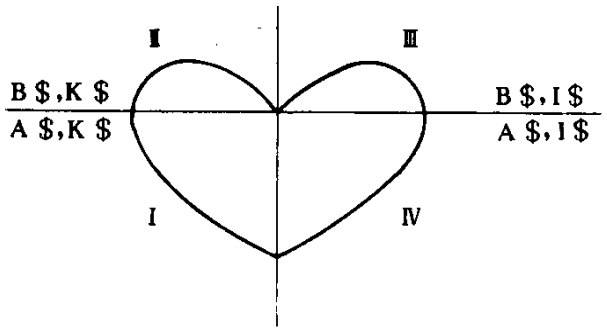


图 5—1

由于 F BASIC 语言没有三角函数,对数和求根指令,所以画曲线比较困难。例 5.10 的程序巧妙地使用了字符处理指令 INSTR、MID$、LEN 函数画出了心形图案。

为便于理解,特对程序说明如下:

程序 No.5—10 共分四部分:

- 20～60 行为初始化程序。
- 100～180 行为绘图子程序。
- 80 行为结束判断处理语句。
- 70、90 两行为绘图控制语句,这两行是本程序的关键行。

程序的执行过程如下:

1. 变量赋值后,通过 70 行第一个 GOSUB 转绘图子程序,画出

心形的左下部分，这时A$、K$参与绘图。(见图5－1Ⅰ)。绘图后返回70行。

2. A$、B$交换内容后，由第二个GOSUB再转绘图子程序，画出心形的左上部分，这时B$、K$参与绘图。(见图5－1Ⅱ)。返回70行，执行90行。

3. K$、I$交换内容，I＝1。返回70行，再由第一个GOSUB语句转绘图子程序，画出心形的右上部分。这时B$、I$参与绘图。(见图5－1Ⅲ)。返回70行。

4. B$、A$交换内容，由第二个GOSUB又转绘图子程序画出心形的右下部分。这时A$、I$有效。绘图后，返回70行，由于I＝1，则执行80行，绘图结束。

关于绘图子程序的分析，读者可以列一张变量变化表，详细列出P、K、C、V、H、X、Y的变化，即可体会出用字符函数绘图的奇妙。

另外，程序中160行使用了符号判断式(K＝2)、(V＝2)，这是一种特殊的判断函数，这种判断式函数共有四种形式，分别为：

X＝A

X＞A

X＜A

X＜＞A

其函数为0、－1。当关系式成立，函数值为－1；当关系式不成立，其函数值为0。用数学式表示(以第一种形式为例)则为：

$$(X=2)=\begin{cases}-1 & \text{当 } X=2 \text{ 时}\\ \text{零} & \text{当 } X<>2 \text{ 时}\end{cases}$$

为使读者加深印象，请看演示程序。

```
10 REM "(X=A)"
20 CLS:INPUT "A=";A
30 INPUT "X=A";X
40 PRINT "("X"="A")="(X=A)
50 PRINT "("X">"A")="(X>A)
```

```
60 PRINT "("X"<"A")="(X<A)
70 PRINT "("X"<>"A")="(X<>A)
80 PRINT "IF CHANGE 'A' PRESS 'Y'";A$=INKEY$(0)
90 IF A$="Y" THEN 20
100 PRINT:GOTO 30
110 END
RUN    ↙    显示
A=?    键入  5   ↙   显示
X=?    键入  3   ↙   显示
(3=5)=0
(3>5)=0
(3<5)=-1
(3<>5)=-1
IF CHANGE 'A' PRESS 'Y'"
```

提示"如果改变 A 值则按 Y 键"。

我们不改变 A 值，按除 Y 键外的任一键。

```
X=?    键入  5    则打印
(5=5)=-1
(5>5)=0
(5<5)=0
(5<>5)=0
X=?    键入  6    则打印
(6=5)=0
(6>5)=-1
(6<5)=0
(6<>5)=-1
.....
```

可多运行几遍，以加深对判断函数的认识。

5.3 出错处理指令

在编制一个复杂的程序时，不可能把程序运行的各个方面都

考虑周到。尤其在游戏程序中，游戏过程千变万化，可能会出现越界、溢出等错误。当出现错误时，一般情况下，计算机要停止运行，打印出错提示。只有人工排除错误后，重新启动，才能继续运行。这种情况若在工业生产自动控制系统中，出现的错误得不到及时纠正，必然要影响生产的正常进行；若在游戏过程中，刚刚玩出一个好的成绩，却因出错中断，游戏再从头开始，也必然令人扫兴。因此，要求程序有自动纠错能力。当运行过程中发生错误时，计算机能够自动进行相应的处理，从而保证程序的正常运行。

F BASIC 语言共有程序出错处理指令五条分别为出错转移语句(ON ERROR GOTO)、错误处理返回语句(RESUME)、出错语句行测试语句(ERL)、出错代码测试语句(ERR)、和自定义出错代码语句(ERROR)。这五条语句，在 F BASIC 语言中，尚未有任何资料公布。下面逐一介绍。

5.3.1 出错转移语句(ON ERROR GOTO)

格式：ON ERROR GOTO ＜行号＞

类型：语句

简写：O. ERR. G.

功用：设置出错转移入口，以便对错误进行“拦截”。该语句也称为陷阱设置语句。

F BASIC 在运行过程中出现的错误有两类：一类是致命性错误；一类是非致命性错误。致命性错误无法补救，只能修改程序、重新运行。而非致命性错误不须中断程序的正常运行，可在运行过程中进行补救。ON ERROR GOTO 语句就是为补救第二类错误而设计的。

如果程序没有执行 ON ERROR GOTO 语句，当遇到错误时，要中止运行，并给出出错信息。而使用 ON ERROR GOTO 语句之后，程序不用中断就可以转移到出错处理子程序中，对错误进行处理。待处理完毕后，再返回到程序的某一位置，继续正常的运行。

注释:ERROR 为出错或错误的意思,“ON ERROR GOTO……”是当程序出错时,转去执行……的意思。

行号是出错处理子程序的首行行号。

当程序在执行期间出现错误时,ON ERROR GOTO 语句就会转向这个行号所规定的子程序。

ON ERROR GOTO 语句须在预计的错误发生之前执行,否则无效。习惯上写在程序的开始处。

说明:1. 同一程序中可以使用几个 ON ERROR GOTO 语句,后面的 ON ERROR GOTO 语句会使前面的 ON ERROR GOTO 语句失效。

2. 如果 ON ERROR GOTO 语句中的行号为 0,则关闭或取消前面的 ON ERROR GOTO 语句的“拦截”功能。

3. ON ERROR GOTO 语句对出错处理子程序中发生的错误不能捕获。若出错处理子程序发生错误,程序将停止运行,显示出错信息。若要对出错处理子程序中的错误进行俘获,则出错处理子程序本身还要再设 ON ERROR GOTO 语句及出错处理子程序。如果不再对错误进行捕获,可在出错信息处理程序内增设 ON ERROR GOTO 0 语句。

示例:

```
10 REM “ON ERROR GOTO”
20 ON ERROR GOTO 100
30 INPUT “X=”;X
40 A=100/X
50 PRINT “100/”X“=”A
60 GOTO 20
100 PRINT:PRINT “Division by 0 !”:PRINT“ERROR”
110 END
RUN    ↙    显示
X=?    键入  7    则打印
```

100/7=14

X=?　　键入　0　　则打印

Division by 0 !

ERROR

意思是说,“用 0 做除数! 出错。”

5.3.2 错误处理返回语句(RESUME)

格式:RESUME <NEXT/行号>

类型:语句

简写:RESU.

功用:通知 F BASIC 从出错处理子程序中返回并恢复执行原调用程序。

如果程序中设置了 ON ERROR GOTO 语句,则当程序出错时,计算机就会令 F BASIC 转去执行出错处理子程序。当错误处理完毕后,F BASIC 就需要返回到调用程序。RESUME 语句就是为返回调用程序而设计的。

注释:RESUME 是恢复或重新开始的意思。

如果 RESUME 之后选用了 NEXT,程序就会在出错语句的下一语句恢复运行。此时 RESUME 与 RETURN 作用类似。如果选用了行号,则程序将会返至标号所标识的语句,并从那里继续运行。

如果行号与 NEXT 都省略,程序就会转到原出错语句。但是,如果错误没有得到改正,RESUME 可能会导致程序进入到死循环。

省略行号和 NEXT 与行号为 0 是等价的。

示例:

```
10 REM "ON ERROR GOTO-RESUME"
20 ON ERROR GOTO 100
30 INPUT "100/X=? INPUT X=";X
40 A=100/X
50 PRINT "100/"X"="A
60 GOTO 20
```

```
100 PRINT "Division by 0 ERROR! INPUT A GAIN!"
110 RESUME 30
120 END
RUN  ↙
100/X=? INPUT X=? □ 键入 12  ↙
100/12=8
100/X=? INPUT X=? □ 键入 0  ↙
Division by 0 ERROR! INPUT AGAIN!
100/X=? INPUT X=?
.....
```

可见，使用了 ON ERROR GOTO……RESUME 语句后，计算机自动处理错误，保证了程序的正常运行。

若把上例中 110 行的行号的 30 去掉，将会看到反复打印："Division by 0 ERROR! INPUT AGAIN!"，程序陷入了死循环。

5.3.3 出错语句行号的测试(ERL)

格式：ERL

类型：系统变量

功用：给出最近一次出错语句所在的行号。

注释：ERL 源于 Error Line，为错误语句行的行号。

示例：

```
PRINT ERL
```

该语句执行后将显示出错误语句行的行号。

示例：

```
10 REM "ERL"
20 CLS:ON ERROR GOTO 100
30 INPUT "INPUT X=";X
40 A=X*X*X
50 PRINT X"∧3="
60 PRINT:GOTO 30
100 PRINT "ERL="ERL
```

```
110 RESUME 30
120 END
RUN      ↙      显示
INPUT X=? □ 键入  9      ↙      打印
9∧3=729
INPUT X=? □ 键入  300      ↙      打印
ERL=40      出错行号为 40
INPUT X=? □
.....
```

5.3.4 出错代码的测试(ERR)

格式:ERR

类型:系统变量

功用:给出最近一次出错语句的出错代码。

注释:ERR 源于 Error,为错误或出错的意思。

每一类错误都有一个确定的代码,错误的类型不同,出错代码也不同。当程序没有错误时,ERR=0。如某一类错误发生,ERR 则记录下这个错误的类型代码。当又有新的错误发生时,ERR 又会变成新的错误代码。

F BASIC 语言共有 22 个错误代码,编号分别为 0～21。表 5—1 给出各出错代码。

表 5—1 错误代码表

错误代码	错误表示	错误信息	说明
0	NF	NEXT Winthout FOR	虽无 FOR 指令,却有 NEXT
1	SN	Syntax error	语法错误
2	RG	RETURN Without GOSUB	虽无 GOSUB,却有 RETURN
3	OD	Out of DATA	READ 读的资料,未备于 DATA 中。

错误代码	错误表示	错误信息	说明
4	IL	Illegal function call	函数的称呼法错误。
5	OV	Overflow	运算结果超过许可的范围。
6	OM	Out of memory	内存不足
7	UL	Undefined Line Number	无GOTO、GOSUB、IF等所指定的行号
8	SO	Subscript out of range	数组变量的下标在规定外
9	DD	Duplicate Definition	数组变量设双重定义
10	DZ	Division by zero	0不能做除数。
11	TM	Type mismatch	变量的类型不一致
12	ST	String too long	字串超过31个字符
13	FT	Formula too complex	式子过于复杂。如()太多
14	CC	Cant't continue	CONT无法再继续执行程序
15	UF	Undefind Function	使用了未定义的函数
16	MO	Missing operand	指令所需要参数未指定
17	TP	Tape read ERROR	无法从卡带正确地读出资料
18	NR	No RESUME	错误处理子程序无RESUME
19	RE	RESUME without error	无错误却遇到RESUME
20	NB	No BGGET	无BGGET却遇到BGPUT

错误代码	错误表示	错误信息	说明
21	UP	Unprintable error	遇到无法显示打印的错误代码

所代表的错误信息及简要说明。

了解每种错误代码所代表的错误类型后，就可在程序运行出错时转去执行相应的错误处理子程序。

5.3.5 自定义出错代码(ERROR)

格式：ERROR n

类型：语句

简写：ERR.

功用：由用户自行定义自己的出错代码。

注释：ERROR 在英文中为错误的意思。

在程序设计中，有时要根据出错类型做相应的转移处理。但往往有些人为设定的错误不在 F BASIC 系统设定的 22 种错误之中。如输入的数值不能大于 1000 或者不能等于 500 等等。这时可使用 ERROR 语句定义用户自己的出错代码，以作为程序判断、转移、处理的依据。

n 为一整数，取值范围为－32768～＋32767。它的有效取值范围为 0～21。当 n 的取值大于 21、小于 256 时，计算机自动设定 n 恒等于 21；当 n 的取值小于零或大于 255 时，计算机自动设定 n 恒等于 4。

示例：

```
IF X=100 THEN ERROR 12
```

例 5.12 编程演示五条出错处理指令的应用和作用。

```
10 REM "ERROR etc。"
20 CLS:ON ERROR GOTO 100
30 INPUT "Please Input";X
40 IF X>1000 THEN ERROR 10
```

```
50 IF X=500 THEN ERROR 15
60 IF X<100 THEN ERROR 20
70 PRINT "OK-OK":GOTO 30
100 IF ERR=10 THEN PRINT "X>1000"
110 IF ERR=15 THEN PRINT "X=500"
120 IF ERR=20 THEN PRINT "X<100"
130 PRINT "ERL="ERL,"ERR="ERR
140 RESUME 30
150 END
RUN  ↙  显示
Please Input ? □   键入 1234,打印
X>1000
ERL=40 ERR=10
(表示出错行号为 40,出错代码为 10。)
Please Input ? □   键入 500,则打印
X=500
ERL=50 ERR=15
Please Input ? □   键入 200,则打印
OK-OK
Please Input ? □   键入 50,则打印
X<100
ERL=60 ERR=20
Please Input ? □   键入 325,则打印
OK-OK
Please Input ? □ ……
```

5.4 程序运行跟踪指令

在程序的调试过程中经常采用跟踪技术,跟踪程序的运行,以了解程序执行的中间结果,查出程序中存在的错误。跟踪指令有两

条，即开始跟踪指令(TRON)和结束跟踪指令(TROFF)。

5.4.1 开跟踪指令(TRON)

格式：TRON

类型：语句或命令

简写：TRO.

功用：对程序的运行位置进行跟踪。

在跟踪状态下，程序在执行过程中会不断显示正在执行着的语句的行号。

注释：TRON 取自于 Trace on，为打开跟踪的意思。

TRON 须与 TROFF 配合使用。后者可在不须跟踪时用于关闭跟踪。

示例：

```
10 REM"TRON"
20 TRON
30 PRINT "TRON"
40 GOTO 60
50 REM "Do Not!"
60 PRINT "Use TRACE"
70 PRINT "TRACE END"
80 TROFF
90 PRINT "It doesn't TRACE"
100 TRON
110 PRINT "TRACE AGAIN"
120 END
RUN    ↙    打印
#30 TRON              (自 30 行开始跟踪)
#40#60Use TRACE       (40、60 行用于跟踪)
#70 TRACE END         (70 行跟踪结束)
#80 It doesn't TRACE  (80 行以后没有跟踪)
#110 TRACE AGAIN      (110 行始又被跟踪)
```

#120
OK

说明:TRON 若作为程序语句使用可只跟踪某一段程序;若作为命令使用可以对整个程序进行跟踪。

5.4.2 关跟踪指令(TROFF)

格式:TROFF

类型:语句或命令

简写:TROF.

功用:关闭 TRON 语句或命令对程序的跟踪功能。

当 TRON 执行后,程序就会在执行过程中不断显示正在执行的语句行号。当欲结束跟踪时,就可用 TROFF 关闭跟踪。

注释:TROFF 取自于 Trace off,为关闭跟踪的意思。

示例:

```
10 REM"TRON—TROFF"
20 TRON
30 PRINT "Trace begin!"
40 FOR I=0 TO 100
50 TROFF:PRINT "Trace off!"
60 PRINT I" * " I"=" I * I,
70 NEXT
80 TRON
90 PRINT "The END!"
100 END
RUN     ↙      显示
#30 Trace begin!  #40#50 Trace off!
0 * 0=0          Trace off!
1 * 1=1          Trace off!
.....
99 * 99=9801  Trace off!
100 * 100=10000#90
```

```
The END!
#100
```

5.5 外设控制指令

人们把一个程序通过键盘输入计算机，当运行或工作完毕后，通常要把计算机电源关掉。在关掉电源的同时，计算机内存中的程序也随即消失了。若再想运行这段程序，必须重新从键盘上逐句输入，这是相当麻烦的事情。为了解决这一问题，人们引入了外存储设备。关机前，先把程序存入外存储设备，下次再使用这段程序时，只要发一个装载指令，再从外存储设备中取回就可以运行，既方便又可靠。外存储设备有多种，常用的外存设备是磁盘机或磁带机，最廉价的磁带机就是录音机。我们的学习机中已设计了专用的录音机接口，供存储和读取磁带程序。

磁带程序的存取指令有三条，分别为：存储指令(SAVE)、存储核对指令(LOAD?)和取出指令(LOAD)。

5.5.1 存储指令(SAVE)

格式：SAVE ["文件名"]

类型：命令

简写：SA.

功用：将指定的程序资料存入录音机的磁带中。

注释：SAVE 的英文意思是存储。

存储程序时，要首先给程序起一个名字，称为文件名，当以后欲使用这一程序时，只要通过指定的文件名，输入 LOAD 指令即可将程序从磁带中取出装入内存。

对于把程序存入磁带的过程，就录音机而言是写入，就计算机而言则是输出。同样，对从磁带中取出程序的过程，就录音机而言是输出，就计算机而言就是装入。两种说法是等价的，读者要习惯

这种说法。

注意文件名必须用双引号引起来。

示例：

```
SAVE "TEST"  ↙      (写入 TEST)
WRIGING TEST        (提示正在写入 TEST)
OK                  (写入完毕)
```

5.5.2 核对指令(LOAD?)

格式：LOAD? ["文件名"]

类型：命令

简写：LO.？或 LO. P.

功用：核对待保存的程序是否已正确地写入到磁带中去。

程序写入录音机磁带后，是否正确写入，一般要进行核查校对。若写入的程序无误则可放心关机，若写入的程序有差错，执行 LOAD? 命令后，会给出有错提示，这时必须把程序另写一遍，以保证资料的完整性。

注释：LOAD? 源于 LOAD PRINT，在此表示对磁带上的内容进行核对。

当指定文件名时，则在找到所确定的文件名的程序之前，会跳过其它程序寻找。

当省略文件名时，则核对的是最初找到的文件。

示例：

```
LOAD?  ↙      (无文件名的情况)
LOADING       (在核对中)
OK            (核对正确)
□
LOAD? "TEST"  ↙      (有文件名的情况)
LOADING TEST
OK
□
```

? TP ERROR　　（核对结果有错）

OK

□　　请重新写入

5.5.3 取出指令(LOAD)

格式:LOAD ["文件名"]

类型:命令

简写:LO.

功用:将程序由录音磁带中取出装入到系统内存中

注释:LOAD 为载入的意思。

文件名为用 SAVE 命令保存程序资料时使用的名称。

说明:

1. 用 LOAD 指令从录音磁带输入程序资料前,应先将内存中的程序清除。

2. 无论输入 LOAD 或 LOAD?,都应在按回车键后,同时按录音机的放音键。

示例:

LOAD　↙　　（不带文件名时则输入第一个文件）

LOADING TEST

OK

□

LOAD"TEST－2"　↙　　（带文件名时）

SKIP TEST　　（提示跳过 TEST 文件）

SKIP TEST－1　　（提示跳过 TEST－1）

LOADING TEST－2　　（提示正在装入 TEST－2）

OK　　（装入完毕）

5.5.4 磁带程序的存取操作过程

1. 写入过程

按使用说明书要求联接好信号线和做好录音机的准备工作

后，首先对欲存储的程序起一个名字，如“GAME1”，然后键入

SAVE“GAME1”

启动录音机后（录音机处于录音状态），按回车键。这时，闪烁光标消失，显示

WRITING GAME1

意思为正在写入文件“GAME1”。过一段时间显示

OK

□

表示程序存储完毕。停止录音。

2. 核对过程

把磁带倒回到开始录音的位置，键入核对命令

LOAD? “GAME1” ↙

按下录音机的放音键（PLAY 键）。这时，光标消失，随后显示

LOADING GAME1

表示正在核对“GAME1”。过一段时间，若出现

OK

□

表示写入正确，可清除“GAME1”程序，再输入欲写入的第二个程序（如 GAME2），重复上述操作。再写入第三个、第四个……。

若在核对过程中，显示

? TP ERROR

OK

□

则表示写入的程序有错误，要进行重写，写后再核对，直到正确写入为止。

3. 读出过程

若磁带中存有许多程序，如 GAME1、GAME2、GAME3……等。现在我们要读出的程序名为“GAME2”，可如下操作。

首先按说明书联接好信号线，磁带倒在适当的位置（一般从头

开始)，录音机处于放音准备状态，然后键入

LOAD "GAME2" ↙

按下录音机的放音键(PLAY)，过一会儿，显示

SKIP GAME1

表示跳过"GAME1"，再过一会儿显示

LOADING GAME2

表示正在装入"GAME2"程序，直到出现

OK

□

表示程序装入完毕。可用 LIST 命令列出程序清单，检查读出的程序是否正确。若显示

? TP ERROR

OK

□

表示无法从磁带中读出程序，可重读，直到正确地读出程序。

4. 如何正确地读出磁带程序

一般来说，把程序写入磁带较容易，这可以通过扬声器听到刺耳的吱吱声和嘎嘎声。但从磁带中读出资料则比较困难。主要原因是：一方面要求录音机的放音频响好；另一方面是有的学习机对输入信号电平要求较苛刻，如科特 FCS－90 一体化游戏学习机，要求输入信号电平在 0.8V 左右，稍高、稍低都不能正确输入。读者可从以下几个方面着手解决：

(1)选用放音质量较好的，音调频率分级调整的录放机。调整音调电位器，衰减低音，提升高音。

(2)仔细调整音量旋钮的位置以选择合适的输出信号电平。要反复试验，找出读取程序成功的最佳位置。

(3)具备无线电常识的读者可尝试在录音机放大器的输入端加入高通滤波器，拆除放大器的高频衰减器，以改善放大器的高频特性。具体参数可根据放大器的特性试验确定。

(4)若上述方法不能奏效，可考虑在录音机与计算机之间加一个简单的接口电路，电原理图如下(见图 5—2)：

注：RL：录音机扬声器的阻抗，一般选 4～8Ω。D1、D2 为普通整流二极管 2CZ、2CP、1N400 等型号都可以，要求正向压降接近 0.8V 若读者没有挑选二极管正向压降的仪器，可选用图 5—3 的电路。使用时仔细调整电位器 W 使 V_0 接近 0.8V 即可。

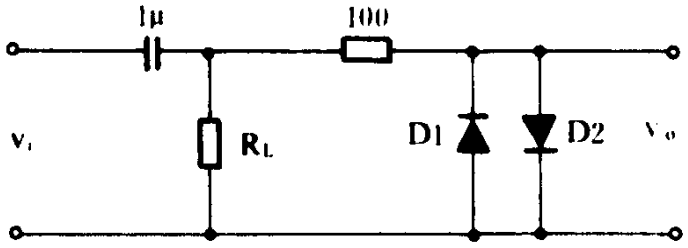


图 5—2　装入接口电路原理图(a)

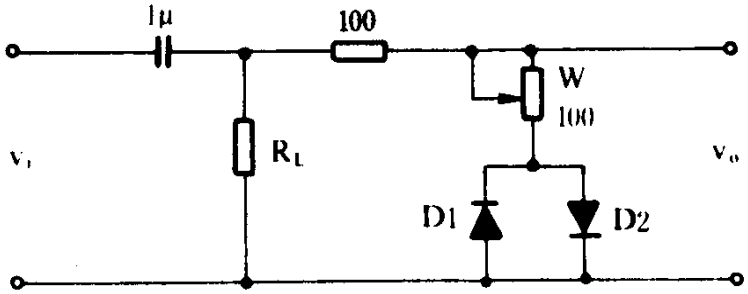


图 5—3　装入接口电路(b)

接口电路的输入端可接 Φ3.5mm 的插座，输出端接 Φ3.5mm 的插头。这样可把原插入计算机 LOAD 口的插头插入接口电路的插座，而把接口电路的输出插头插入计算机。(采用插座插头的目的是为了避免接口电路输入、输出端混淆而损坏录音机放大器。)

采用上述电路后，只要仔细调整音调及音量电位器，一般可以成功。

5.5.5　置监控指令(SYSTEM)

格式：SYSTEM

类型:命令或语句

简写:S.

功用:将计算机置于监控方式。

注释:SYSTEM 在英文中为系统的意思,在此指监控系统。

示例:

```
SYSTEM    ↙    显示
 SYSTEM       ON
```

意思是系统打开。

说明:由于 F BASIC 没有监控程序的解释程序,所以监控系统打开后,计算机处于等待状态,貌似死机。可按 STOP 键返回 F BASIC,这时相当于进行了一次热启动。

由于 SYSTEM 指令在直调 BASIC 状态下不能发挥作用,所以不多介绍。

5.6 功能键控制指令

我们学习机的键盘上一般都有八个功能键,编号为 F1～F8。实际上,键盘上的每个键都可以说是功能键,如数字键"9"可在屏幕上显示数字 9,并把 9 输入显示缓存区;同样字母键"A"可以把字母 A 显示于屏幕上。这些键的功能是由系统的解释程序确定的,用户不能随意改变。因此,这些键一般不称其为功能键。而 F1～F8 这八个键的功能可以由用户随意定义,它们对用户是开放的。用户可以把它定义为一个语句、一个单词、几个符号甚至是一句话。如把 F1 定义为"SAVE",那么当需要把程序写入磁带时,就不必连续键入 S、A、V、E 四个字母键而直接按 F1 键就可输入"SAVE"命令。可见通过合理地定义功能键的内容可加速程序的输入和方便键盘的操作。

5.6.1 显示功能键定义指令(KEYLIST)

格式:KEYLIST

类型:命令或语句

简写:K. L.

功用:将功能键的定义内容显示于屏幕上。

在功能键的使用中,当各键的定义较陌生时,可能短时间内记不准确,可通过 KEYLIST 指令列出各功能键的定义以利使用。

注释:KEYLIST 为键列表的意思。

示例:

```
KEYLIST    ↙      打印
F1 GAME0(M)       (M)表示按 RETURN 键
F2 GAME1(M)
F3 GAME2,1(M)
F4 GAME3(M)
F5 SPRITE
F6 PRINT
F7 LIST
F8 RUN(M)
```

这是 V3. 10 F BASIC 系统原定义的八个功能键的内容。

5. 6. 2 设定功能键指令(KEY)

格式:KEY N,"字符串"

类型:命令或语句

简写:K.

功用:对某一功能键进行重新定义

当键盘输入或操作使用的单词或语句较多时,可使用 KEY 指令把单词或语句定义给某一功能键,以提高程序输入速度或简化操作。

注释:KEY 在此为键的意思。

N 的取值为 1～8。

字符串可以是除双引号之外的任意字符,一个功能键最多可

以定义 15 个字符。定义的字符串必须用双引号括起来。

示例：

```
KEY 1,"FCS－90"  ↙
```

这时按 F1 键则屏幕显示 FCS－90。

例 5.13 在对话 BASIC 状态下，编程显示功能键原定义内容，更改 F1～F4 内容，再显示新定义后各键的内容。

```
10 REM "No. 5－12 KEY－KEYLIST"
20 CLS:PRINT "OLD DEFINATIONS"
30 KEYLIST
40 KEY 1,"DEF SPRITE"
50 KEY 2,"POSITION"
60 KEY 3,"CLS:SPRITE ON"
70 KEY 4,"DEF MOVE ("
80 PRINT "NEW DEFINATIONS:"
90 KEYLIST
100 END
RUN  ↙  打印
OLD DEFINATIONS:
F1 LOAD(M)
F2 PRINT
F3 GOTO
F4 CHR$(
F5 SPRITE
F6 CONT(M)
F7 LIST(M)
F8 RUN(M)
NEW DEFINATIONS:
F1 DEF SPRITE
F2 POSITION
F3 CLS:SPRITE ON
```

```
F4 DEF MOVE (
F5 SPRITE
F6 CONT(M)
F7 LIST(M)
F8 RUN(M)
OK
□
```

在原功能键的定义中有 RUN(M)、CONT(M)等指令后有(M)标志，它表示相当于按回车键，即按该功能键后不按回车键就可以执行。但是当我们按这种形式定义功能键时却不能自动执行。这种功能键的定义方法如下：

如欲定义立即执行的清屏命令

```
CLS(M)
```

可键入

```
KEY 1,"CLS"+CHR$(13)
KEYLIST      ↙      则打印
F1 CLS(M)
F2 ……
……
OK
□
```

这时键入 F1，则立即清屏。

定义功能键命令中的"CHR $(13)"即相当于按回车键。(13即为回车键的十进制键值。)

5.7 音响处理指令

在 BS 状态，通过人机对话可以进入 MUSIC BOARD(谱曲与演奏)状态。在该状态下，我们可以使用 F1～F7、1～7、A～G21 个

键谱制四个八度音的三重奏乐曲，乐曲为钢琴音色，七种演奏速度可调，音色优美动听。但遗憾的是 MUSICBOARD 仅有一页的版面供使用，这一页仅有四行，最多可演奏 96 个音符，且只有一种音色，音符的时值变化范围也较窄（一般仅能取四分音符和八分音符），这对于跌宕起伏、节奏强烈、音色多变的乐曲简直就不能演奏。更有甚者，MUSICBOARD 每次只能演奏一小段，一首乐曲要断续几次甚至十几次才能演奏完毕，因而破坏了乐曲的美感和完整，影响了乐曲的欣赏效果。另外在游戏程序中，伴随着游戏的进行要加进不同的音响或音乐，这些仅靠 MUSICBOARD 也显然不能实现。因此要想欣赏优美的乐曲，编制富有吸引力的游戏程序就必须使用音响处理指令，才能达到目的。

F BASIC 语言的音响处理指令有三条：一条是键控发声指令（Ctrl＋G），可控制扬声器发出 1000Hz 的蜂鸣声；一条是编程发声指令（BEEP），发声效果与 Ctrl＋G 相同；另一条是音乐演奏指令（PLAY）。使用 PLAY 指令可以演奏具有八种节拍变换、四种音色效果、16 节音量、音长变化、6 个八度音、72 键钢琴旋律的三重奏乐曲。也只有使用 PLAY 指令，才能把我们带入电脑演奏乐曲的美妙境界。

5.7.1 键控发声指令（Ctrl＋G）

格式：Ctrl＋G

类型：命令

功用：控制扬声器发出 1000Hz 的蜂鸣声。

5.7.2 程控发声指令（BEEP）

格式：BEEP

类型：语句或命令

简写：B.

功用：控制扬声器发出 1000Hz 的蜂鸣声。

注释：BEEP 源于 BEE POP，是蜂鸣声的意思。

BEEP 语句可以在游戏程序中使用(比如作为达到或击中某一目标的回应),也可以在程序的某个地方用来设计报警信号(比如用来提示输入错误等)。

示例

```
10 REM "BEEP"
20 INPUT "CONTINUE (Y/N)?";A$
30 IF A$ ="Y" THEN 20
40 IF A$ ="N" THEN 100
50 BEEP
60 PRINT "PLEASE PRESS 'Y' OR 'N' KEY!"
70 GOTO 20
100 END
RUN ↙
CONTINUE (Y/N) ? □      询问继续操作吗?
    键入"Y"则继续打印
CONTINUE(Y/N)? □      若键入 A ↙
    (扬声器响一声)      打印
PLEASE PRESS 'Y'OR 'N'KEY!
    提示"请按 Y 或 N 键!"
CONTINUE(Y/N)? □      键入 N ↙
OK
□      停机。
```

例 5.14 编程演示 BEEP。

```
10 REM "No. 5—13 BEEP"
20 FOR I=0 TO 10
30 BEEP:PAUSE 10
40 NEXT
50 PAUSE
60 GOTO 20
70 END
```

RUN ↙

(扬声器断续响 11 次)　　出现一闪烁光标

□　　按任一键

(扬声器又断续响 11 次)　　又出现一闪烁光标。

□……

5.7.3 音乐演奏指令(PLAY)

格式:PLAY "弦乐器资料"

类型:语句或命令

简写:PL.

功用:令扬声器播放或演奏乐曲。

注释:PLAY 为播放或演奏的意思。

弦乐器资料为特定的字符串,它规定了音程、8 度音、音量、和音、主音、音色效果等资料,使用冒号可以演奏三重音。

示例:

PLAY "CRDRE" ↙

则演奏 1 0 2 0 3 0

说明:

1. 弦乐器资料约定

(1)节拍

节拍以 T1～T8 指定

T1(快)←→T8(慢)

如 PLAY"T3"

(2)音色

音色以 Y0～Y3 指定

Y0:12.5%

Y1:25.0%

Y2:50.0%

Y3:75.0%

要在指定 M0 或 M1 后，再指定音色 Y0～Y3。

(3)主音

主音由 M0、M1 指定。

M0：表示音量，以 V0～V15 指定音量的高低。

V0(最低)←→V15(最高)

如：PLAY "T3M0V4Y1"

指定节拍为 3、音量为 4、音色为 25%

M1：表示音长，以 V0～V15 指定主音的长度。

V0(最短)←→V15(最长)

如：PLAY "T3M0V4Y1M1V7"

指定主音长度为 7。

(4)8 度音

8 度音以字母 O 指定，O0～O5。

以简谱中 1 音为例，O0～O5 对应的 8 度音如下：

O0　O1　O2　O3　O4　O5

1　1　1　1　1　1

若乐曲中有 1 音，可将 8 度音升高一级或几级。如

O0　O1　O2　O3　O4　O5

1　1　1　1　1　1

(5)音名和音符

音名分别用 C、D、E、F、G、A、B 表示。

音符分别用 1、2、3、4、5、6、7 表示。

音名和音符的对应关系如下：

全音：

C　D　E　F　G　A　B

1　2　3　4　5　6　7

半音：

#C　#D　#F　#G　#A

1#　2#　4#　5#　6#

如：

PLAY "T4M0V3M1V5Y2O3D"

指定节拍为 4、音量为 3、音长为 5、音色为 50.0%、第 3 号 8 度音、音名为 D，发音为"2"。

(6)休止符

休止以 R 表示，R 后面缀以 0～9 的整数，以表示休止时间的长短。

R0(最短)←→R9(最长)

休止的确切时间见音符时值的约定。

(7)音符的时值

在音名的后面附上 0～9 的整数用来指定音名的时值，时值与整数的对应关系见表 5—2，仍以"1"音为例。

表 5—2

数字	音符	拍节	示　例
9	全音符	4 拍	1—— —— ——
8	付 2 分音符	3 拍	1—— ——
7	2 分音符	2 拍	1—
6	付 4 分音符	$1\frac{1}{2}$拍	1·
5	4 分音符	1 拍	1
4	付 8 分音符	3/4 拍	$\underline{1}$·
3	8 分音符	1/2 拍	$\underline{1}$
2	付 16 分音符	3/8 拍	$\underline{\underline{1}}$·
1	16 分音符	1/4 拍	$\underline{\underline{1}}$
0	32 分音符	1/8 拍	$\underline{\underline{\underline{1}}}$

如：

```
10 PLAY "T3M0V9M1V5Y3"
20 PLAY "O3C3O2A3O3C3D3E7"
```

RUN ↙ 则演奏

|1 6 1 2 3—|

(8)重音

F BASIC 可以演奏二重奏和三重奏乐曲。演奏重音时可在字符串中间加冒号实现。演奏重音的格式为：

PLAY "系统 A:系统 B"

PLAY "系统 A:系统 B:系统 C"

如：

PLAY "O3C5:O2E5:1G5" ↙

则发出 1、3、5三音的重奏。

2. 语法结构和编程约定

(1)一般先指定主音的音量和音长 M0、M1，然后再指定节拍 T、音色 Y。各参数一经指定后会一直保持到再次指定时为止。若未指定时，系统会自动设定各参数初值为"M0V15T4O3"、且音符的时值为 5。

(2)每行语句都必须使用 PLAY 指令引导，字符串用双引号括起来。每个 PLAY 指令引导的字符不能超过 31 个。在二、三重奏时，冒号也作为一个字符对待。

(3)在定义二、三重奏时，必须先定义节拍 T，几个系统的节拍必须一致，否则重奏不能和谐同步。

(4)音符的时值一经指定后，若后一个音符的时值与前一个一样，后一音符可省略时值，直到需改变时值时再指定，即使换行也不影响。

如 2 3 2 1|7 6 5 |可编写为

"O3D3EDCO2BAC5"而不必写为

"O3D3E3D3C3O2B3A3C5"

(5)每句中的 8 度音只要后一个与前一个音符的相同，可不必再指定，直到重新指定 8 度音为止，即使换行也不影响。

如上例中不必写为

"O3D3O3EO3DO3CO2BO2AO2C5"。

(6)在三重奏时,系统C的音色是固定不变的。因此系统C可省略音色的指定。

(7)在二、三重奏中,每个系统必须分别指定。各系统的音色、音量、音长、音符的时值等参数一经指定,可在本系统内从上一语句转移到下一语句。

3. 编程方法

(1)首先把乐谱以拍分段,分段距离稍大些,一般十六开纸竖用每行分六至八段。若是重奏,也要对应分好。

(2)在每个乐谱下面填上对应的音名,然后再对照表5—2填上每个音名的时值,若后一音名的时值与前一音名的时值相同时,后一音名可以省略时值。

(3)在音名前面填上8度音On,若后一音名的8度音与前一音名相同,则后一音名可以省略8度音。

当掌握熟练后,2、3两步可一次完成。

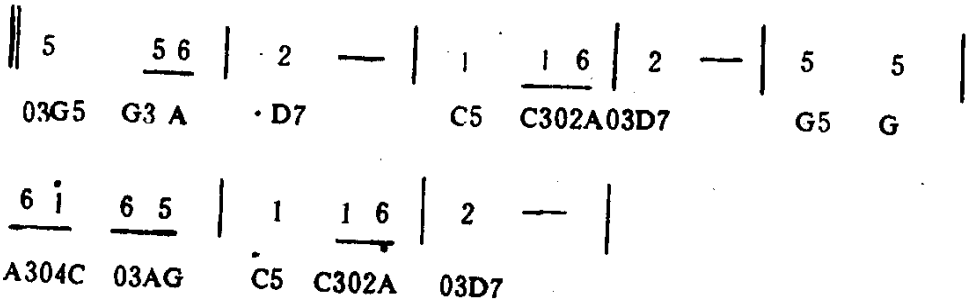
(4)将字符按照每行少于或等于31个字符的规则填入PLAY语句。注意,在分行时一拍之内的音名字符不能分开。

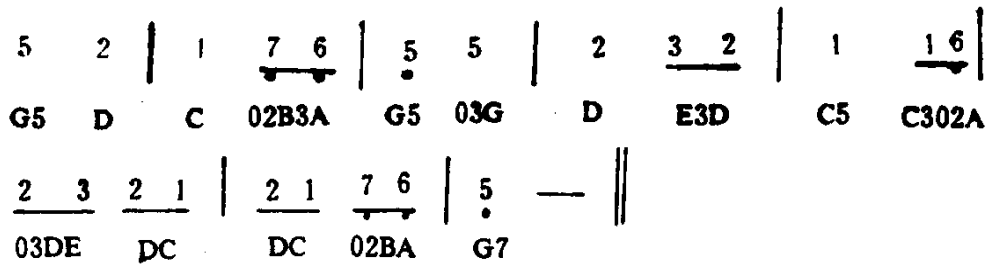
(5)输入计算机,运行,修改错误,满意后,写入录音磁带,以备使用。

例5.15 编制演奏《东方红》乐曲的程序。

首先分段写出乐谱(取分行6段),填上音名等。

The EAST IS RED





填好后核对无误，填入 PLAY 语句。

本例设定“M1V15Y2T4”

程序如下：

```
10 REM “No. 5－14 The East is red”
20 PLAY “M1V15Y2T4”
30 PLAY “O3G5G3AD7C5C3O2AO3D7”
40 PLAY “G5GA3O4CO3AGC5C3O2AO3D7”
50 PLAY “G5DCO2B3AG5O3GDE3DC5C3O2A”
60 PLAY “O3DEDCDCO2BAG7”
70 END
RUN ↙
```

演奏东方红乐曲。

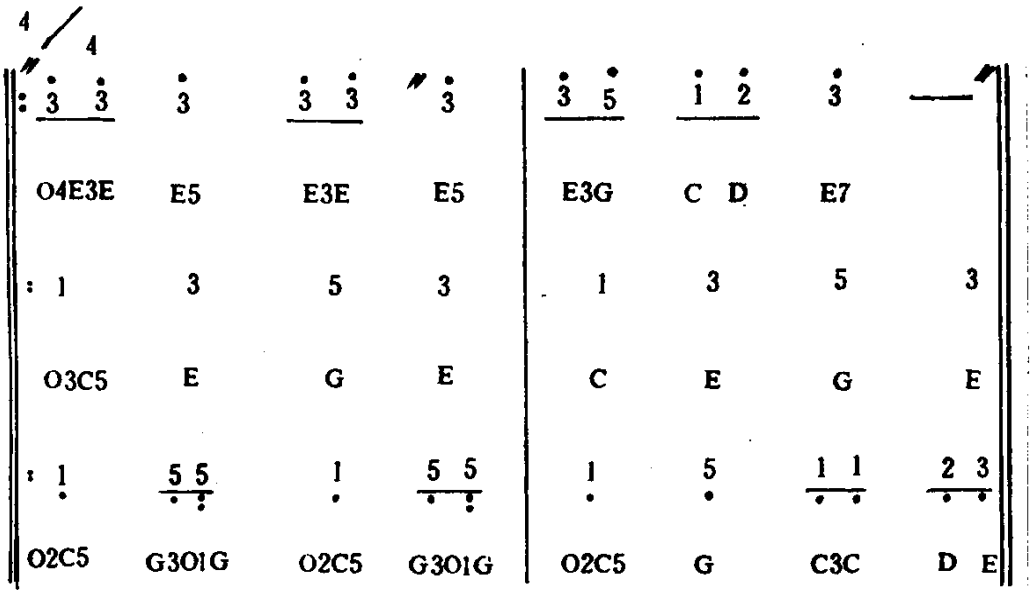
下面再列举一个三重奏的乐曲编程示例。

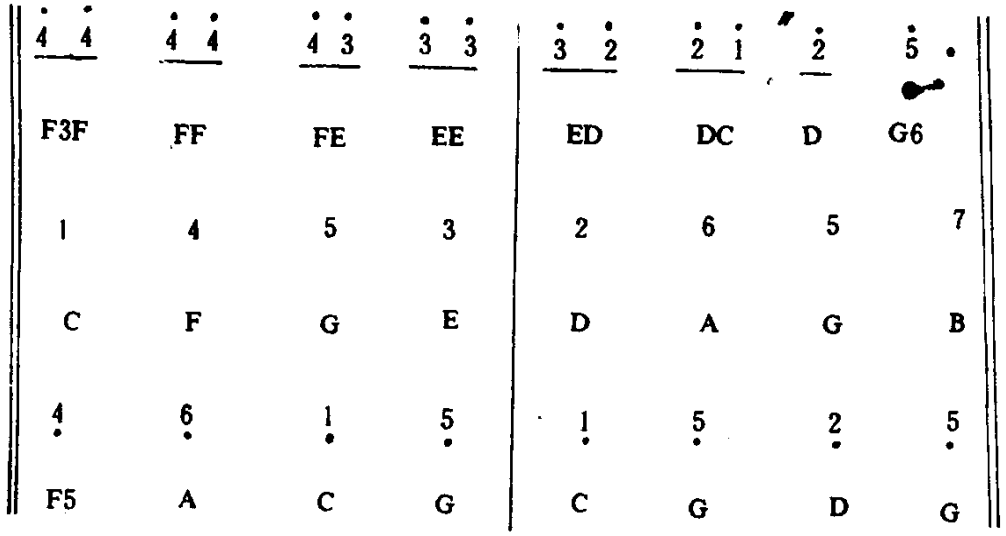
例 5.16 编写《铃儿响叮铛》的三重奏乐曲。

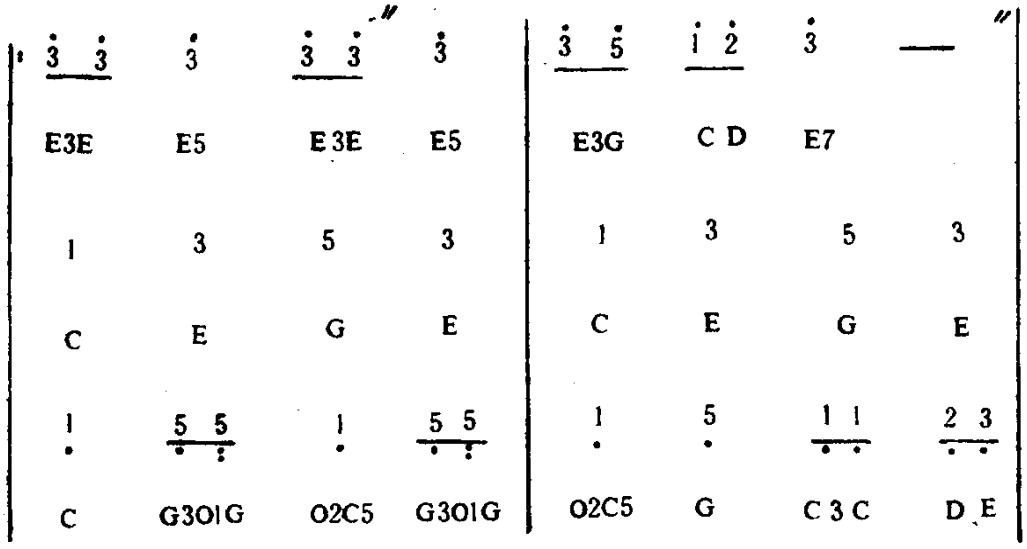
《铃儿响叮铛》是彼尔彭特谱写的世界名典，现将原曲改编成三重奏曲。

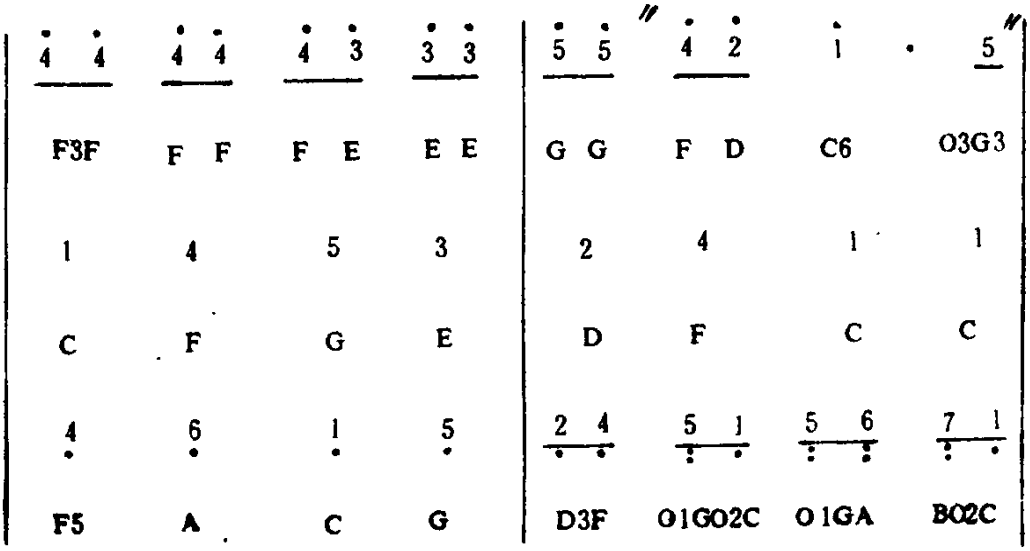
首先分段列出三重奏曲谱(本例分八段)，然后依次填入音名、时值、8 度音等。核对无误后，便可填入 PLAY 语句中。由于是三重奏曲，要求每个 PLAY 语句中的字符数小于或等于 29 个，因此，在填入语句前要先对字符分句。分句方法是：按节拍逐拍计算三个系统的字符之和，以小于或等于 29 个字符为一句。但要注意二分音符和付四分音符的情况，必须把该音符所在的节拍及后一个节拍填在一句中。如上例的第八拍、第十六拍、第二十四拍、第三十二拍、第四十拍、第四十八拍、第五十六拍、第六十四拍，还有第三十

六拍、第四十四拍、第五十二拍。否则，将出现节拍紊乱。






3 3 | 3 | 3 3 | 3 | 3 5 | 1 2 | 3 | —
E3E | E5 | E3E | E5 | E3G | C D | E7 |
1 | 3 | 5 | 3 | 1 | 3 | 5 | 3
C | E | G | E | C | E | G | E
1 | 5 5 | 1 | 5 5 | 1 | 5 | 1 1 | 2 3
C | G3O1G | O2C5 | G3O1G | O2C5 | G | C 3 C | D E


4 4 | 4 4 | 4 3 | 3 3 | 5 5 | 4 2 | 1 | 5
F3F | F F | F E | E E | G G | F D | C6 | O3G3
1 | 4 | 5 | 3 | 2 | 4 | 1 | 1
C | F | G | E | D | F | C | C
4 | 6 | 1 | 5 | 2 4 | 5 1 | 5 6 | 7 1
F5 | A | C | G | D3F | O1GO2C | O1GA | BO2C

5 3	2 1	5	· 55 //	5 3	2 1	6	· 6 //
G04E	D C	03G6	G1G	G304E	D C	03A6	A3
1	3	5	3	1	3	4	2
C5	E	G	E	C	E	F	D
1	5	1 5	5 5	1	5	4 6	6 6
C5	G	C301G	G G	02C5	G	F301A	A A

6 4	3 2 //	6	· 6 //	5 5	4 2 //	3	· 5
A04F	E D	C3A6	A3	04GG	F D	E6	03G3
2	6	4	2	3	2	3	1
D	A	F	D	E	D	E	C
4	6	4 6	6 6	5 5	6 7	1 5	5 1
02F5	A	F301A	02A01A	02G01G	A B	02C01G	G02C

5 3	2 1	5 ·	5	5 3	2 1	6 ·	6
GO4E	D C	O3G6	G3	GO4E	D C	O3A6	A3
1	3	5	3	1	3	4	2
C	E	G	E	C	E	F	D
1	5	1 5	5 5	1	5	4 6	6 6
C5	G	C3O1G	G G	O2C5	G	F3O1A	A A

6 4	3 2	5 5	5 5	6 5	4 2	1	—
AO4F	E D	G G	G G	A G	F D	C 7	
6	4	7	5	4	2	1	—
A	F	B	G	F	D	C 7	
4	6	5	7	5 5	6 7	1	—
O2F5	A	G	B	G3O1G	A B	O2C7	

程序如下：

```
10 REM "No. 5－15.JINGLE BELLS"
20 PLAY "M1V9Y2T3;MOV5M1V5YOT3;T3"
30 PLAY "O4E3EE5E3E;O3C5EG;O2C5G3O1GO2C5"
40 PLAY "E5E3GCDE7;ECEGE;G3O1GO2C5GC3CDE"
50 PLAY "F3FFFFEEEEDDC;CFGEDA;F5ACGCG"
60 PLAY "DG6E3EE5E3E;GBCEG;DGCG3O1GO2C5"
70 PLAY "E5E3GCDE7;ECEGE;G3O1GO2C5GC3CDE"
80 PLAY "F3FFFFEEEGG;CFGED;F5ACGD3F"
90 PLAY "FDC6O3G3;FCC;O1GO2CO1GABO2C"
100 PLAY "GO4EDCO3G6G1G;C5EGE;C5GC3O1GGG"
110 PLAY "G3O4EDCO3A6A3;CEFD;O2C5GF3O1AAA"
120 PLAY "AO4FED;DA;O2F5A"
130 PLAY "O3A6A3;FD;F3O1AO2AO1A"
140 PLAY "O4GGFD;ED;O2GO1GAB"
150 PLAY "E6O3G3GO4E;ECC;O2CO1GGO2CC5"
160 PLAY "DCO3G6G3GO4E;EGEC;GC3O1GGGO2C5"
170 PLAY "DCO3A6A3;EFD;GF3O1AAA"
180 PLAY "AO4FEDGGGG;AFBG;O2F5AGB"
190 PLAY "AGFDC7;FDC7;G3O1GABO2CC7"
200 END [或 GOTO 20]
```

核对无误后，输入计算机，就可以欣赏这首世界名曲，演奏效果比真正的钢琴还要优美动听。

若音乐编程熟练后，可直接根据乐谱输入程序，那么其输入速度并不比 MUSICBOARD 慢，而效果要比它强好多。

5.8 进制转换指令

5.8.1 16→10 进制转换

在本章第二节介绍 VAL 函数时，我们曾讨论了如何使用 VAL

函数把 16 进制数转换成 10 进制数。实际上使用输出指令 PRINT 也能把 16 进制数转化成 10 进制数。

格式:PRINT &H(16 进制数)

类型:命令

功用:把 16 进制数转化成 10 进制数。

示例:

```
PRINT &H F      ↙      立即显示
15
PRINT &H AB     ↙      显示
171
PRINT &HFABC    ↙      显示
-1348
PRINT &H8FAC    ↙      显示
-28756
```

5.8.2 10→16 进制转换

格式:HEX $ (X)

类型:函数

简写:H.

功用:将某个十进制整数转换成相应的十六进制数。

注释:HEX 源于 Hexadecimal,为十六进制数的意思。

X 为被转换的十进制数,取值范围为-32768～+32767 它可以是变量或表达式。

示例:

```
PRINT HEX $ (255)       ↙      显示
FF
PRINT HEX $ (-28756)    ↙      显示
8FAC
PRINT HEX $ (-32768)    ↙      显示
8000
```

PRINT HEX $(32767)　↙　显示

7FFF

通过示例可见，F BASIC 系统中十进制、十六进制之间的对应关系与数学中的对应关系不同。它们之间的对应规律如表 5—3

希望读者注意这一特殊的对应关系。

表 5—3　F BASIC 10—16 进制数对照表

十六进制数据	十进制数据	十六进制数据	十进制数据
0000	0	8000	−32768
01FF	511	C000	−16384
03FF	1023	E000	−8192
07FF	2047	F000	−4096
0FFF	4095	F800	−2048
1FFF	8191	FC00	−1024
3FFF	16383	FE00	−512
7FFF	32767	FFFF	−1

5.9　空余内存测试指令(FRE)

在把程序调入或输入内存时，为防止程序太长超过内存容量而丢失数据，要首先了解内存空余单元数目。使用 FRE 函数可解决这一问题。

格式：FRE

类型：函数

简写：FR.

功用：报告计算机中尚未被程序占用的空余字节数(即剩余空间)。

注释：FRE 取自于 Free 一词，在计算机中通常为未用空间或

自由空间的意思。

示例：

PRINT FRE ↙ 显示

6353

键入 NEW ↙

PRINT FRE ↙ 显示

8182

调入 GAME 0 游戏程序，按 STOP 键

PRINT FRE ↙ 显示

6840

可见，FRE 函数的值是随用户使用内存的情况而变化的。我们的学习机在直调 BASIC 状态，最大自由空间为 8182(BYTES)；在对话进入 BASIC 状态，最大自由空间为 4030(BYTES)。编程时，要根据程序量的大小选用恰当的状态，以保证完整的装入程序。

5.10 返回 BS 菜单指令(SYSTEM)

在直调 BASIC 状态，指令 SYSTEM 用于进入监控系统，由于 F BASIC 没有监控解释程序，所以 SYSTEM 指令无用。但在对话 BASIC 状态，SYSTEM 则有另外的作用。

格式：SYSTEM

类型：命令或语句

简写：S.

功用：从 F BASIC 执行状态返回 F BASIC 选择菜单。

注释：SYSTEM 在此表示返回主控程序(菜单)的意思。

我们知道在对话进入 F BASIC 状态时出现如下选择菜单。

F BASIC

1— —BASIC

2— —BG GRAPHIC

3— —END

当我们选择 1 时则进入 BASIC 编程状态，在编制程序中，当我们想进入 BG GRAPHIC 状态进行键控绘图时，可键入

SYSTEM ↙

即可返回 F BASIC 选择菜单，选择 2，即进入 BG GRAPHIC。当绘图完毕，再进入 BASIC 状态时，可以发现绘图前编制的程序完好的保存下来。所以，使用 SYSTEM 命令具有保护原程序的作用。

5.11 滤色指令(FILTER)

格式：FILTER <n>

类型：语句

简写：FIL.

功用：改变底背景的色调。

使用 FILTER 语句可对 PALETB 指令选定的底背景颜色调色。

注释：FILTER 在英文中为滤色器，滤光镜的意思。

n 的取值范围为 0～7，可以是数值常数、变量或表达式。

FILTER 具有加强底背景某种色调的作用，取不同的 n 值，加强不同的颜色，其对应情况见表 5—4。

表 5—4 FILTER 滤色对照表

n	对应的颜色	n	对应的颜色
0	灰白	4	兰
1	绿	5	青
2	红	6	紫
3	橙	7	黄

示例：

FILTER 3 ↙

```
OK
□
FILTER  7    ↙
OK
□
```

例 5.17 编程演示 FILTER 语句的八种变化。

```
10 REM "No. 5－16 FILTER"
20 CLS
30 PALETB 0,52,33,39,43
40 FOR I=0 TO 7
50 PAUSE
60 FILTER I
70 BEEP
80 LOCATE 0,20:PRINT I
90 NEXT
100 GOTO 40
```

RUN ↙ 出现一闪烁光标。每按一次空格键，色彩改变一次，并打印出 FILTER 语句中的 n 值。可见，FILTER 语句相当于在底背景的投影机上安装了一块滤色镜。

5.12 调用机器语言子程序指令(CALL)

格式：CALL <X>

类型：命令或语句

简写：CA.

功用：调用机器语言子程序，并向子程序传递参数。

机器语言程序具有运行效率高，占用内存少等优点。CALL 指令为 F BASIC 使用机器语言子程序提供了可能。

注释:CALL 为调用的意思。

X 为机器语言子程序的入口地址,取值范围为－32768～＋32767。它可以是数值常数、变量或表达式。

说明:关于机器语言子程序的编写,我们将另文讨论。下面仅给出在 V3.10 版本中某些子程序的作用(供参考)。

1. 色调控制子程序:

(1)CALL －19400:可使底背景变为绿色。

(2)CALL(－19401)或者(－19404)或者(－19398):作为命令使用可把显示方式置为黑底绿字;作为语句使用可置显示方式为浅兰色底背景、粉红色字符。

(3)CALL －19397 或(－19394):作为命令使用置显示为浅黄色底背景、深蓝色字符。作为语句使用可置深蓝色底背景、粉红色字符。

(4)CALL －19392:作为命令时置淡黄色底背景。作为语句时置蓝色底背景、白色字符。

(5)CALL －18950(或－18940):可使字符变为金黄色。

2. 关显示子程序:

CALL －19396 可关闭屏幕显示开关,使屏幕变为全黑。

3. 图案翻滚子程序:

CALL －19000(或者－19002)可使画面向上翻滚,旋转显示。犹如画面绘制在一个圆筒上,圆筒旋转,画面反复显示。

4. 旋转打印:

CALL －19006(或－19008)可使光标在圆筒上旋转打印,而打印的画面不动。

5. 彩色边界上移:

(1)CALL －19310(或－19311～－19314、－19298、－19930、－19940)可在屏幕两边打印出彩色边界,每边两个字符 16 点,它们占据高分辨图形域,故当中留出 28 个字符的空白。若垂直打印,

可令两边界飞速上移。

示范程序如下：

```
10 CALL -19310
20 PRINT " "
30 GOTO 20
40 END
```

(2)仅打印左边界：

CALL－19303(或－19304)可只在屏幕左边打印边界

6. 随机打印彩色字符、图案：

(1)CALL－19306(或－19308)可在屏幕上随机打印各种彩色字符、图案。

(2)CALL－19960 仅在屏幕左边 1/3 的范围内随机打印各种图案。

7. 清除内存和复位

CALL－19999(或－19998～－19990)可清除内存和系统复位。具有 NEW 和复位的双重作用。

8. 设置显示方式：

CALL－19391(或－19389、－19249、19235 等)可置 CGEN 0。

读者可对自己的电脑版本使用 CALL 指令逐个单元试验，找出可供调用的子程序。

5.13 参考指令

到此为止，我们已介绍了 F BASIC 语言的 104 条指令，其中有 74 条是 F BASIC 语言的基本指令，剩余的是扩展指令。

我们知道 F BASIC 语言是由其它版本的 BASIC 语言扩展、移植而来的，如天马、大岛、BIT－79 键盘使用的 F BASIC 语言是由日本 NS－HU BASIC 语言移植而来，而科特键盘的 F BASIC 语言

则是由日本 MSX BASIC 语言移植而来。由于参照的版本不同，移植产生的 F BASIC 语言也有许多版本，如天马、BIT－79 键盘使用 F BASIC V3.1 版本，大岛键盘则使用 F BASIC V3.3 版本，而科特 FCS－90 键盘在对话 BASIC 使用 F BASIC BS.2A 版本，在直调 BASIC 则使用 F BASIC V3.10 版本。

由于 F BASIC 语言版本之间的差异，致使各种键盘能够识别、执行的指令数量产生了较大的差别。一般来说，对于 74 条基本指令，各种 F BASIC 版本都能识别执行。对于扩展指令，有些版本能够识别执行，有些版本只能部分识别执行，有些版本只能识别不能执行，有些版本则根本就不能识别，更不能执行。请读者注意区别。

为使读者能够较全面系统地了解 F BASIC 语言的全貌，本节再介绍 8 条参考指令，这些指令在不同的版本上可能有不同的格式和功用，下面仅介绍一般的使用格式，读者可在此基础上对自己使用的版本进行探讨。

5.13.1 BACKUP

格式：BACKUP n1[,n2]

类型：命令

简写：BA.

功用：对磁盘进行备份。即把一个磁盘上的信息全部复制到另一张磁盘上去。

注释：BACKUP 为备份或后援的意思。

n1 为源盘号(即信息源磁盘所在驱动器编号，n2 为目的盘号(即作为备份磁盘所在驱动器编号)。

当只有一个驱动器时，省略 n2，n1 取零。

说明：

为追求售价低廉，目前国内生产、销售的游戏电脑键盘都没有配置磁盘驱动器的接口，所以没法使用磁盘驱动器。因此，BACKUP 指令也暂无用处。

5.13.2 BGGET

格式:BGGET

类型:命令或语句

简写:BGG.

功用:将在 BG GRAHPIC 状态的键控绘图存入 RAM 备用。

注释:BGGET 源于 Background Get,为接收背景的意思。

使用 BGGET 指令可在系统中预存一幅画面,以便在游戏中更换背景时使用。存储方法是:进入 BG GRAPHIC 状态,键控绘图。绘图完毕返回 F BASIC 状态后,键入 BGGET 命令回车后即可存入。

5.13.3 BGPUT

格式:BGPUT

类型:命令或语句

简写:BGP.

功用:将 RAM 中所存储的图象再现到屏幕上。

注释:BGPUT 源于 Background Put,为放回背景的意思。

在游戏中,当欲更换游戏画面时可使用 BGPUT 指令把存入系统中的画面调出显示。

5.13.4 TAB

格式:TAB(n)

类型:函数

简写:TA.

功用:按列对数据进行定位输出。

注释:TAB 取自 Table,为表或列表的意思。

n 取值在 0～255 之间,可取数值常量、变量或表达式。

说明:TAB 函数一般用在 PRINT 语句,有的版本还可用在 INPUT 语句中,如

```
PRINT TAB(3);"A";TAB(10);"B"
```

```
INPUT TAB(5)"PLEASE INPUT A=";A
```

TAB 函数中的 n 值是从屏幕首列字符的位置开始计算的，因此同一语句中有几个 TAB(n)时，n 值要从左至右逐一递增。

5.13.5 SPC

格式：SPC(n)

类型：函数

功用：使输出数据跳过若干个横格，确定后面所跟的输出项的输出位置。

注释：SPC 取自于 Space，为空格的意思。

n 的取值范围为 0～255，可以取数值常数、变量或表达式。

与 TAB 函数不同，SPC 函数的 n 值是从光标的当前位置开始计算的，所以当 n 值大于 27 时，F BASIC 会自动进行求余处理。即令 n MOD 27=n。

5.13.6 CLICK ON

格式：CLICK ON

类型：语句

简写：CLI. O.

功用：令按键时发出卡嗒声。

注释：CLICK 在英文中为卡嗒声和一拍即合的意思。

5.13.7 CLICK OFF

格式：CLICK OFF

类型：语句

简写：CLI. OFF

功用：关闭按键时的卡嗒声。

小　结

关于 F BASIC 语言的指令介绍，到此已全部结束。包括 CLICK 指令在内，本书共介绍了 111 条指令。这些指令已基本上满足了编

制运算程序，事务处理程序和游戏程序的需要。但是F BASIC语言是BASIC大家族中的新成员，是一个新生事物。同一切新生事物一样，F BASIC语言也有一个被人们逐渐认识、了解、熟悉、掌握的过程。

但是F BASIC语言决不仅限于这111条指令，如画面的纵向翻滚，画面的横向卷动，画面卷动时两幅画面的衔接等等效果。还有如何把游戏中的主人翁拼凑成更大的卡通，如32×32点或64×32点等操作，都要有相应的指令控制处理。尤其是随着F BASIC版本的不断更新，将会推出更多、更有效的指令。愿我们共同发挥自己的聪明才智，深入探讨发掘，使F BASIC语言在我国的微机普及教育事业中开出璀璨的花朵。

为便于读者掌握使用F BASIC语言，特列出两个版本的F BASIC指令汇总表(见表5—5、表5—6)。

表 5—5　F BASIC BS. 2A 版指令汇总表

ABS	KEY	RETURN
ASC	LET	RIGHT $
BEEP	LEN	RND
CALL	LEFT $	RUN
CGEN	LIST	SAVE
CGSET	LINPUT	SCR $
CUT	LOAD	SGN
CHR $	LOAD?	Shift＋HOME
CLEAR	LOCATE	SPC
CLS	MID $	SPRITE
COLOR	MOVE	SPRITE n
CONT	MOVE(n)	SPRITE OFF
CSRLIN	NEXT	SPRITE ON
Ctrl＋E	NEW	STEP
Ctrl＋G	ON—GOSUB	STICK
Ctrl＋L	ON—GOTO	STOP
Ctrl＋Z	ON—RETURN	STRIG
DATA	ON—STORE	STR $
DEF MOVE	PALETB	SWAP
DEF SPRITE	PALETS	SYSTEM
DIM	PAUSE	TAB
ERA	PEEK	THEN
FOR	POKE	TO
FRE	PLAY	VAL
GOSUB	POS	VIEW
GOTO	POSITION	XPOS
HEX $	PRINT	YPOS
IF	READ	
INPUT	REM	
INKEY $	RESTORE	

表 5-6　F BASIC V3. 10 版指令汇总表

ABS	DELETE	MOVE
AND	DIM	MOVE(n)
ASC	ERA	NEXT
AUTO	ERL	NEW
BACKUP	ERR	NOT
BEEP	ERROR	ON ERROR GOTO
BGGET	FILTER	ON—GOSUB
BGPUT	FIND	ON-GOTO
BGTOOL	FOR	ON—RETURN
CALL	FRE	ON—STORE
CAN	GAME	OR
CGEN	GOSUB	PALETB
CGSET	GOTO	PALETS
CHR $	HEX $	PAUSE
CLEAR	IF	PEEK
CLICK ON	INPUT	PLAY
CLICK OFF	INKEY $	POKE
CLS	INSTR	POS
COLOR	KEY	POSITION
CONT	KEYLIST	PRINT
CRASH	LET	READ
CSRLIN	LEN	REM
Ctrl+E	LEFT $	RENUM
Ctrl+G	LIST	RESUME
Ctrl+L	LINPUT	RESTORE
Ctrl+Z	LOAD	RETURN
CUT	LOAD?	RIGHT $
DATA	LOCATE	RND
DEF MOVE	MID $	RUN
DEF SPRITE	MOD	SAVE

续表 5—6

SCR $	STEP	TAB
SCREEN	STICK	TROFF
SGN	STOP	TRON
Shift+Home	STRIG	VAL
SPC	STR $	VCT
SPRITE	SWAP	VIEW
SPRITE n	SYSTEM	XPOS
SPRITE OFF	THEN	XOR
SPRITE ON	TO	YPOS

F BASIC 的 V3.1、V3.3 版本同 V3.10 版本所能识别的指令能力差不多，读者可参考表 5—6。

5.14 程序绘图初探

在 BG GRAPHIC 状态键控绘图，虽然直观简便，容易实现，但这种方法不能占据屏幕的下三行，从而使图形占用屏幕的面积减少百分之十二，屏幕使用效率低。

实际上应用最多的还是程序绘图。所谓程序绘图就是通过编写程序绘制图案。但是 F BASIC 语言没有专门的绘图指令，欲编程绘图只有通过 LOCATE 语句控制图形打印位置而绘出各种图案。本节简要介绍几个简单程序的绘图实例，系统的介绍留待下篇编程技巧中讨论。

5.14.1 米字形图案

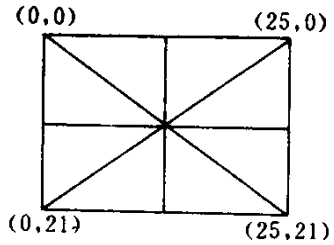
编程绘出米字形图案，米字外面用方框围起来。

程序如下：

```
10 REM "No. 5—17"
20 CLS
```

```
30 FOR X=0 TO 25
40 LOCATE X,0:PRINT " * "
50 LOCATE X,21:PRINT " * "
60 LOCATE X,10:PRINT " * "
70 LOCATE X,(21-21 * X/25):PRINT " * "
80 LOCATE X,21 * X/25:PRINT " * "
90 NEXT
100 FOR Y=0 TO 21
110 LOCATE 13,Y:PRINT " * "
120 LOCATE 0,Y:PRINT " * "
130 LOCATE 25,Y:PRINT " * "
140 NEXT
150 END
```

RUN ↙ 画出图案如下：



程序中 30～90 行完成了三条横线和两条斜线的打印；100～140 行完成了三条竖线的打印。

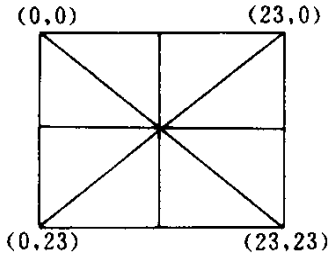
若调整图形的比例和尺寸，可以用一次循环画出图形

调整后的绘图程序如下：

```
10 REM "No. 5--18"
20 CLS:SCREEN 1,0
30 FOR I=0 TO 23
40 LOCATE 0,I:PRINT " *  *  * "
50 LOCATE I,0:PRINT " * ":LOCATE I,12:PRINT " * ":LOCATE I,23:
   PRINT " * "
```

```
60 LOCATE I,I:PRINT "*"
70 LOCATE 23-I,I:PRINT "*"
80 NEXT
90 END
RUN ↙
```

画出图案如下：



程序中 40 行打印三条竖线，50 行打印三条横线，60、70 行打印两条斜线。对比之下，第二个程序要简洁一些。

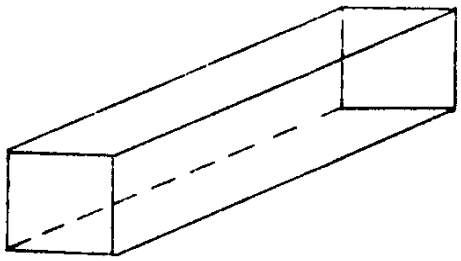
5.14.2 长方体图案

绘制长方体图案，实质上就是先绘制两个坐标不同的正方形，然后再用四条斜线把两个正方形的四角连起来。绘图关键是计算好四条斜线的斜率。程序如下：

```
10 REM "No. 5-19"
20 CLS
30 FOR X=0 TO 9
40 LOCATE 0,X:PRINT "*"
50 LOCATE X,0:PRINT "*"
60 LOCATE 9,X:PRINT "*"
70 LOCATE X,9:PRINT "*"
80 NEXT
90 FOR X=17 TO 26
100 LOCATE X,9:PRINT "@"
110 LOCATE X,18:PRINT "@"
120 NEXT
130 FOR Y=9 TO 17
```

```
140 LOCATE 17,Y:PRINT "@"
150 LOCATE 26,Y:PRINT "@"
160 NEXT
170 FOR X=1 TO 16
180 LOCATE X,9*X/17:PRINT CHR$(205)
190 LOCATE X,9*X/17+9:PRINT CHR$(205)
200 NEXT
210 FOR X=10 TO 25
220 LOCATE X,9*X/17-4:PRINT CHR$(205)
230 LOCATE X,9*X/17+5:PRINT CHR$(205)
240 NEXT
250 END
```

RUN ↙ 打印图形如下



为便于理解，No. 5—19 全部使用了单循环，可采用多重循环以简化程序，留给读者练习。

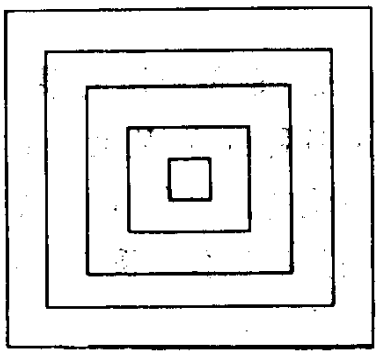
5.14.3 正方形套圈

绘制不同尺寸的五个正方形，正方形的中心在同一个点上。

编程如下：

```
10 REM "No. 5-20"
20 CLS:X=13:Y=11:A$=CHR$(253)
30 FOR N=1 TO 10 STEP 2
40 FOR L=-N TO N
50 LOCATE X+L,Y-N:PRINT A$
```

```
60 LOCATE X+N,Y+L:PRINT A$
70 LOCATE X-L,Y+N:PRINT A$
80 LOCATE X-N,Y-L:PRINT A$
90 NEXT:NEXT
100 END
RUN  ↙   打印如下图形
```



No. 5－20 使用于双重循环，使程序简洁清新，读者可自行分析编程的技巧。

若在 No. 5－20 中再加入一段程序，使方框从里圈开始陆续显示，待显示出全部图形后，再从外圈开始陆续消失，全部消失后再陆续显示。如此周而复始，产生动感，留给读者自己练习。

第六章　数学运算程序的编写技巧

如何打破整数BASIC取值范围的限制，象浮点BASIC那样方便地进行较大数值的各种运算，甚至优于浮点BASIC而进行各种高精度计算是本章讨论的重点和目的。怎样达到这一目的呢？我们通过考察一个简单的加法运算实例引出解决这一问题的途径。

例　求43725＋44132的和。

在笔算中一般是列竖式计算。即

$$\begin{array}{r} 43725 \\ +\quad 44132 \\ \hline 87857 \end{array}$$

计算结果是87857。归纳起来，计算步骤共三步：

1. 写出两个加数，各位对齐。

2. 对应位分别相加，结果写在对应位的下方。

3. 把每位相加的结果，按高位在前低位在后的顺序组成一个新的数字。

显然，上例的计算在F BASIC语言系统中用赋值语句或输出命令PRINT是不能进行的。

我们知道，F BASIC语言系统变量中有一种变量——下标变量。由同一变量名的下标变量组成的集合叫数组。在上例的计算中，如果引进A(5)、B(5)、C(5)三个数组，将两加数的每一位分别装入A、B数组中，然后将A、B数组对应单元的内容分别相加，相加之和存入C(5)数组的对应单元中，那么C数组各变量中的内容

就是两数相加的和。由于进行的运算是单位数的相加，所以不会有溢出问题。具体计算步骤如下：

1. 将两数分别装入 A(5)、B(5)数组中：

A(1)=5　A(2)=2　A(3)=7　A(4)=3　A(5)=4

B(1)=2　B(2)=3　B(3)=1　B(4)=4　B(5)=4

2. A、B 两数组各元素内容对应相加，结果存入 C(5)数组中。即

C(1)=A(1)+B(1)=5+2=7

C(2)=A(2)+B(2)=2+3=5

C(3)=A(3)+B(3)=7+1=8

C(4)=A(4)+B(4)=3+4=7

C(5)=A(5)+B(5)=4+4=8

3. 将 C 数组各元素内容，按高下标在前、低下标在后的顺序排列输出，即得结果

C(5)、C(4)、C(3)、C(2)、C(1)=87857

比较两种运算过程，运算顺序一致，运算结果相同，证明这种方法是可行的。由于每个下标变量装入一位数字，下标变量的最大取值为 255，所以引入数组参加运算，可以处理 250 位以内数字的数学运算。推而广之，若一个下标变量装入二位或三位数字，或者采用双下标变量，则能够处理数字的位数可以扩展到千位以上。这样，即便是天文数字，也可以任意处理。本章将深入浅出、由简到繁、分别介绍任意位数的加法、减法、乘法、除法及乘幂、阶乘等运算程序的编写。同时介绍小数、分数的数学运算程序的编写技巧。最后给出三角函数，对数等较复杂的编程思路和方法。

6.1 基本子程序的介绍

为简化程序的编写过程，我们先熟悉几个带有工具性质的子程序。

6.1.1 如何把数据装入数组

F BASIC 语言要求输入的数字小于 32767，输入大于 32767 的数则显示“? OV ERROR”，拒绝输入。但对输入的字符串，则可以接受 31 个字符。因此，把数字当做字符串输入，最高可达 31 位，这在一般数学运算中，完全可以满足要求。

把数字装入数组，一般有两种方法。一种方法是先告诉电脑待输入数据的位数，然后定义数组下标，利用循环语句把数据装入数组(此方法可装入 255 位以内的数据)。见子程序 6.1：

子程序 6.1：数据装入数组程序(A)

```
10 INPUT "SHU JU 1 DE WEI SHU P=?";P
15 INPUT "SHU JU 2 DE WEI SHU Q=?";Q
20 DIM A$(P),B$(Q)
25 PRINT "SHU JU 1 =?";
30 FOR I=P TO 1 STEP -1
35 A$(I)=INKEY$(0):PRINT A$(I);:NEXT
40 PRINT "SHU JU 2=?";
45 FOR I=Q TO 1 STEP -1
50 B$(I)=INKEY$(0):PRINT B$(I);:NEXT
```

运行子程序 6.1，根据提示，顺序输入数据 1 的位数 P，数据 2 的位数 Q，然后根据提示分别装入数据 1，数据 2，就可把数据 1、2 分别装入 A$(P)、B$(Q)数组中。

子程序 6.1 的设计原理不难理解，技巧在于使用了键盘输入语句 INKEY$，自动读入已确定位数的数据。当输入数据的位数达到确定位数 P、Q 时，自动停止输入。通过使用，可以发现，这种

数据装入方法要先输入数据位数，再输入数据，操作较烦琐。在数据位数小于31位的情况下，人们希望一次输入数据，自动装入数组。第二种方法可实现这一要求。

子程序6.2：数据装入数组程序(B)

```
10 INPUT "SHU JU 1 =?";A$:A=LEN(A$)
15 INPUT "SHU JU 2 =?";B$:B=LEN(B$)
20 DIM C(A),D(B)
25 FOR I=1 TO A
30 C$=MID$(A$,I,1)
35 C(A-I+1)=VAL(C$):NEXT
40 FOR J=1 TO B
45 D$=MID$(B$,J,1)
50 D(B-J+1)=VAL(D$):NEXT
```

子程序6.2说明：

(1)10、15行输入数据1、2，测量数据1的字符长度为A；数据2的字符长度为B。

(2)20行定义C(A)、D(B)数组

(3)25～35行把数据1装入C(A)数组。

(4)40～50行把数据2装入D(B)数组。

子程序6.2由电脑自动判断输入数据位数的多少，从而简化了数据输入操作。

6.1.2 如何进行加法运算

数据装入数组后，即可进行对应位相加运算。相加原则为同下标变量的内容对应相加，相加之和存入E数组中(E数组要先定义)，见子程序6.3。

子程序6.3：相同位数的求和运算(A)

```
100 DIM E(A)
105 FOR I=1 TO A
110 E(I)=C(I)+D(I)
```

```
115 NEXT
```

两数相加，可能出现进位，必须考虑进位的处理。见子程序 6.4。

子程序 6.4：相同位数的求和运算(B)

```
100 DIM E(A+1):X=1
105 FOR I=1 TO A
110 E(X)=C(I)+D(I)
115 E(X)>9 THEN E(X)=E(X)-10:E(X+1)=E(X+1)+1
120 X=X+1 : NEXT
```

程序说明：115 行为进位处理语句，当某位之和大于 9 时，则令本位减去 10，高一位加 1。

实际上可以将两数相加之和存入两加数的任一数组中，从而省去一个存和的数组，以减少内存的占用。这只要把 C(A)数组中的下标调整为 A+1 以存放进位，即可达到目的，参见子程序 6.5。

子程序 6.5：求和运算通用程序

```
20 X=1:DIM C(A+1),D(B)
100 FOR I=1 TO A:C(X)=C(X)+D(I)
105 IF C(X)>9 THEN C(X)=C(X)-10:C(X+1)=C(X+1)+1
110 X=X+1:NEXT
```

读者可能已经发现，子程序 6.3～6.5 只适用于两数据位数相同或者数据 1 的位数多于数据 2 的情况。若数据 2 的位数多于数据 1 时，由于只循环 A 次，将要丢失数据，出现错误。因此子程序 6.3～6.5 设计不完满。改进的方法是在数据装入前，先对两数据的位数进行判断处理，使较大的数据装入 C 数组。(见子程序 6.6)

子程序 6.6：通用数据装入程序

```
10 INPUT "SHU JU 1=?";A$:
   INPUT "SHU JU 2=?";B$
15 A=LEN(A$):B=LEN(B$):IF A<B THEN SWAP A$,B$:
   GOTO 15
20 DIM C(A+1),D(B):X=1
```

```
25 FOR I=1 TO A:C(A-I+1)=VAL(MID $(A$,I,1)):NEXT
30 FOR J=1 TO B:D(B-J+1)=VAL(MID $(B$,J,1)):NEXT
```

程序说明:15 行在测试了数据 1、2 的字符长度后,对 A、B 进行比较,若数据 2 的位数多于数据 1,则令 A$,B$ 交换内容。从而保证 A$ 字符变量存最大的数据。

子程序 6.6 是一个通用数据装入数组子程序,可作为工具使用。

6.1.3 运算结果的输出

运算结束后,运算结果存于 C(A+1)数组中,只要置一循环语句,依次从高下标到低下标取出各下标变量的内容排列成串,即得输出结果。(见子程序 6.7)

子程序 6.7:运算结果输出程序

```
200 IF C(A+1)<>0 THEN A=A+1
205 FOR I=A TO 1 STEP-1
210 PRINT C(I);
215 NEXT
```

程序说明:200 行判断最高位是否有进位,若有进位则先取出 C(A+1)中的数据,以保证输出结果的完整性。

6.1.4 结束处理

结束处理程序一般比较简单。若继续运算,则使用 CLEAR 语句使变量清零,重新运行程序;若结束运算,则进行停机操作。程序如下:

子程序 6.8:结束处理子程序

```
300 PRINT:PRINT "JI XU MA? (Y/N)";
305 INPUT Y$
310 IF Y$<>"N"THEN CLEAR:RUN
315 CLS:LOCATE 10,10:PRINT "GOODBYE!!!":END
```

通过以上 8 个子程序的介绍,我们可以编制完整的运算程序了。

6.2 加法运算程序的编制

6.2.1 整数加法

把以上介绍的子程序 6.5、6.6、6.7、6.8 连接在一起就组成了整数加法运算程序(见程序 6—1)。

```
1 REM "NO.6-1 JIA FA"
5 CLS:INPUT"SHU JU 1=";A$:INPUT"SHU JU 2=";B$
10 A=LEN(A$):B=LEN(B$):IF A<B THEN SWAP A$,B$:
   GOTO 10
15 DIM C(A+1),D(B):X=1
20 FOR I=1 TO A: C(A-I+1)=VAL(MID$(A$,I,1)):NEXT
25 FOR J=1 TO B:D(B-J+1)=VAL(MID$(B$,J,1)):NEXT
30 FOR I=1 TO B:C(X)=C(X)+D(I)
35 IF C(X)>9 THEN C(X)=C(X)-10:C(X+1)=C(X+1)+1
40 X=X+1:NEXT
45 IF C(A+1)<>0 THEN A=A+1
50 FOR I=A TO 1 STEP-1:PRINT C(I);:NEXT
55 PRINT:PRINT"JI XU MA? (Y/N)";:INPUT Y$
60 IF Y$<>"N" THEN CLEAR:GOTO 5
65 CLS:LOCATE 10,10:PRINT"GOODBYE!!!":END
```

6.2.2 小数加法

同整数相比,小数多了小数点和小数部分。因此,一个小数必须把整数和小数两部分分别装入两个数组中,这样,两个小数要使用四个数组。然后整数、小数分别对应相加。最后判断小数的十分位是否有进位。若有,把进位加在整数最低位,把两部分连在一起,中间添上小数点即得计算结果。关于小数加法运算程序,除了程序 6.5 不同外,其它部分大同小异。下面仅给出小数装入数组子程序。

子程序 6.9：小数装入数组程序

```
10 INPUT"SHU JU 1=";A$:A=LEN(A$)
15 FOR I=1 TO A:C$=MID$(A$,I,1)
20 IF C$="."THEN P=I:GOTO 30
25 NEXT
30 Z=P-1:W=A-P:DIM Z1(Z),W1(W)
35 FOR I=1 TO Z:Z1(Z-I+1)=VAL(MID$(A$,I,1)):NEXT
40 FOR J=P+1 TO A:W1(J-P)=VAL(MID$(A$,J,1)):NEXT
45 INPUT"SHU JU 2=";B$:B=LEN(B$)
50 FOR I=1 TO B:D$=MID$(B$,I,1)
55 IF D$="."THEN Q=I:GOTO 65
60 NEXT
65 Y=Q-1:R=B-Q:DIM Z2(Y),W2(R)
70 FOR I=1 TO Y:Z2(Y-I+1)=VAL(MID$(B$,I,1)):NEXT
75 FOR J=Q+1 TO B:W2(J-Q)=VAL(MID$(B$,J,1)):NEXT
```

运行子程序 6.9，可将两小数的整数部分装入 Z1、Z2 数组；小数部分装入 W1、W2 数组。使用子程序 6.9 时要注意 35、40、70、75 行中循环变量初、终值的设置。

有了子程序 6.9，再两次调用求和运算程序和运算结果输出程序并配一结束处理子程序，即可得出小数求和运算程序。不过要注意，在进行小数部分求和时，要从左至右对应位相加；在进行整数部分求和时，要从右至左对应位相加。编程中还要考虑小数部分向整数部分个位数的进位和小数点的打印。该程序编制较容易，留给读者练习。

以上所述程序的设计思路是先把数据装入数组，然后对应下标变量之间求和，最后再由数组输出运算结果。这种程序设计方法，层次分明；段落清楚；阅读方便；容易理解；便于掌握。一般要求按上述方法编制程序。

为展宽视野、开拓思路，使读者较快地掌握编程技巧，登堂入

室、举一反三，下面再介绍另一种风格的程序设计方法。这种方法只定义一个装和的数组，参与运算的两个数据一直作为字符串处理，仅在相加时，才把其变为数据。另外，在运算前，先对两数据进行格式化处理，使整数、小数两部分位数相同，不足部分用空格补齐。该程序的另一特点是模拟笔算过程，列出竖式，并演示进位操作，非常直观逼真。整个程序构思巧妙，富有启迪性。现介绍如下：（本方法只适用运算数据少于 31 位的情况）。

程序 6－2：通用加法运算程序（小数、整数均适用）

```
1 REM"NO. 6-2 TONG YONG JIA FA"
5 CLS:CLEAR:DIM R$(40):INPUT"X=";X$
10 INPUT"Y=";Y$:X1$=X$:Y1$=Y$:U$="
          ":N1=1:N2=1:ML=MR(""表示空格)
15 N1=N1+1:X2$=MID$(X$,N1,1):IF X2$="." THEN N1=
   N1-2:GOTO 25
20 GOTO 15
25 N2=N2+1:Y2$=MID$(Y$,N2,1):IF Y2$="."THEN N2=
   N2-2:GOTO 35
30 GOTO 25
35 IF N2<N1 THEN 50
40 IF N2=N1 THEN ML=N1:GOTO 55
45 X$=LEFT$(U$,N2-N1)+X$:ML=N2:GOTO 55
50 Y$=LEFT$(U$,N1-N2)+Y$:ML=N1
55 N1=LEN(X$)-ML:N2=LEN(Y$)-ML
60 IF N2<N1 THEN 75
65 IF N2=N1 THEN MR=N2:GOTO 80
70 X$=X$+LEFT$(U$.N2-N1):MR=N2:GOTO 80
75 Y$=Y$+LEFT$(U$,N1-N2):MR=N1
80 N=ML+MR:CLS
85 FOR I=1 TO N: B$=MID$(X$,I,1):C$=MID$(Y$,I,1)
90 LOCATE 10+(I-1)*2,2:PRINT B$:LOCATE 10+(I-1)*2,
```

```
   4:PRINT C$:NEXT
95 LOCATE 6,6:PRINT"+":LOCATE 4,8:PRINT"--------
   --------"
100 C=0
105 FOR I=N TO 1 STEP-1
110 A=VAL(MID$(X$,I,1)):B=VAL(MID$(Y$,I,1)):Z=A
    +B+C:C+0
115 IF Z>9 THEN C=1
120 R$(I)=RIGHT$(STR$(Z),1)
130 LOCATE 8+I*2,10:PRINT R$(I)
135 IF I=ML+3 THEN I=I-1:R$(I)=".":GOTO 130
140 IF C=1 THEN LOCATE 5+I*2,6:PRINT C
145 FOR J=0 TO 500:NEXT:NEXT
150 IF C=1 THEN R$(0)="1":LOCATE 7,10:PRINT "1":GOTO
    160
155 R$(0)=""
160 PRINT:PRINT X1$"+"Y1$"=";:FOR I=0 TO N:PRINT R
    $(I);:NEXT
165 CLS:LOCATE 10,20:PRINT"JI XU MA? (Y/N)";:A$=
    INKEY$(0):IF A$<>"N"THEN 5
170 CLS:LOCATE 10,10:PRINT"GOODBYE!!!":END
```

程序说明：

(1)15～30 行确定两数小数点位置。

(2)35～75 行对两数字进行格式化处理，使两数统一格式。如 X$="325.849"，Y$="3.9587"，格式统一后，X$="325.849␣"，Y$="␣␣3.9587"。(注"␣"表示空格)45、50 行格式化整数部分。70、75 行格式化小数部分。格式化后，两个字符串的长度一样，小数点位置相同。

(3)85～95 行列竖式，模拟笔算方式。

(4)100～155 行，运算结果，并演示进位过程。具体过程如下：

从右到左分别取出X$、Y$中对应的一个字符，转化成数字后，赋值给变量A、B(110行)，变量C存放低一位的进位。有进位，C=1；无进位，C=0。令Z=A+B+C，当Z>9时令C=1，否则C=0。再把Z转成字符串，取出最后一个字符送入存和字符串数组对应的R$(I)中、(125行)。然后把它们显示在竖式的对应位置上(130行)，运算结果放在R$(40)的N+1个下标变量中。(N是统一格式后X$的长度，N+1<40)。

(5)160行输出结果。

(6)当运算数字的位数包括小数点大于40时，要相应调整5行中R$的下标变量。

(7)当输入数为整数时，必须在整数后加小数点和一个"0"。如35输入为35.0。

程序6－2运行情况如下：

RUN↙　屏幕左上角显示

X=?　　键入325.849↙

Y=?　　键入3.9587↙

清屏后，显示

```
   325.849
     3.9587
+    1 11
-----------
   329.8077
```

325.849+3.9587=329.8077

JI XU MA? (Y/N)　　闪动光标

键入除"N"键外的任一键，可继续运算。

键入"N"则清屏，显示"GOODBYE!!!"停机。

该程序模仿笔算逼真，是一个较好的加法运算程序。

6.3 减法运算程序的编制

有了加法运算程序的编写技巧和经验，只要分析处理好减法运算与加法运算的几个不同点，则很容易编写出减法运算程序。

1. 考虑差的正、负，当被减数小于减数时差为负值。

2. 解决被减数、减数位数相同时，两数大小的判别问题。

3. 处理好被减数小于减数时的借位。

4. 处理好结果的输出，使显示的差符合一般记数习惯。如

325.11－316＝9.11

若打印为 009.11，虽然结果不错，但看起来别扭，这种情况须避免。

6.3.1 整数减法

下面结合程序 6－3 介绍减法程序的编写技巧。

程序 6－3：整数减法程序

```
1 REM"NO.6-3 JIAN FA"
10 CLS:CLEAR:INPUT "Bei Jian Shu=";A$:INPUT"Jian Shu=";
   B$:PRINT A$"-"B$"=";
15 A=LEN(A$):B=LEN(B$):IF A<B THEN SWAP A$,B$:
   SWAP A,B: PEINT"-";
20 DIM C(A+1),D(B)
25 C(A+1)=1:X=1
30 FOR I=1 TO A:C(A-I+1)=VAL(MID$(A$,I,1)):NEXT
35 FOR J=1 TO B:D(B-J+1)=VAL(MID$(B$,J,1)):NEXT
40 FOR I=1 TO B
45 IF C(X)<D(I) THEN C(X)=C(X)+10:C(X+1)=C(X+1)-1
50 C(X)=C(X)-D(I):X=X+1:NEXT
55 FOR J=B+1 TO A:IF C(X)<0 THEN C(X)=C(X)+10:C(X+
   1)=C(X+1)-1
```

```
60 X=X+1:NEXT
65 IF C(A+1)=0 THEN SWAP A$,B$: PRINT "-";:GOTO 25
70 IF C(A)=0 THEN A=A-1:GOTO 70
75 FOR I=A TO 1 STEP-1:PRINT C(I);:NEXT
80 PRINT:PRINT"JI XU MA?(Y/N)":Y$=INKEY$(0):IF Y$<
   >"N" THEN 10
85 CLS:LOCATE 10,10:PRINT"GOODBYE!!!":END
```

程序说明：

(1)10～35 行的功能是把数据装入数组，若减数位数多于被减数，则打印“－”，并将 A$、B$ 的内容交换，保证位数多的数据放入 A$ 字符变量中，并装入 C 数组。25 行 C(A+1)=1 是给被减数超高位预置“1”，用于两位数位数相同时，判别差的符号。

(2)40～60 行为运算程序。45 行的作用是被减数小于减数时，被减数向高位借“－”，当“＋”后，再进行相减。

(3)55～60 行的作用是当两数相减结束后，判断被减数所剩高位是否为负数。若是负数，则向高一位借“－”当“＋”，使本位大于等于 0。高一位处理完毕，再处理高二位，直到所有剩余高位都处理完毕。如 100000－325 后，千位为“－1”，则通过 55 行千位向万位借“－”当“＋”，使千位变为 9；这时万位又变为“－1”，则万位再向十万位借“－”当“＋”，万位变为 9，十万位变为 0。

(4)当被减数、减数位数相同时，15 行不能判断两数的大小，只有运算完毕，通过 65 行检查，予置超高位的“1”是否被借去。若 C(A+1)≠0，说明被减数大，差为正数；若 C(A+1)=0，说明超高位被借去，被减数小于减数，差为负值。要把 A$、B$ 内容对调，重新运算，输出负值的差。

(5)70～75 行是输出程序。70 行从差的高位向低位依次检查，以保证输出的差前面不带“0”，符合记数习惯。

6.3.2 小数减法

根据加、减运算的四个不同点，只要对通用加法程序稍加改造，即可得出通用减法程序。（见程序 6－4）。

程序 6－4：通用减法程序（小数、整数均适用）

```
1 REM "NO. 4 TONG YONG JIAN FA"
5 CLS:CLEAR:DIM R$(40):U$="                    ":INPUT
  "X=";X$
10 INPUT "Y=";Y$:X1$=X$:Y1$=Y$:N1=1:N2=1:P=R
   =W=C
20 N1=N1+1: IF MID$(X$,N1,1)="." THEN 30
25 GOTO 20
30 N2=N2+1:IF MID$(Y$,N2,1)="." THEN 40
35 GOTO 30
40 IF N2<N1 THEN 55
45 IF N1=N2 THEN P=N1:GOTO 60
50 X$=LEFT$(U$,N2-N1)+X$:P=N2: GOTO 60
55 Y$=LEFT$(U$,N1-N2)+Y$:P=N1
60 N1=LEN(X$)-P:N2=LEN(Y$)-P
65 IF N2<N1 THEN 80
70 IF N1=N2 THEN R=N1:GOTO 85
75 X$=X$+LEFT$(U$,N2-N1):R=N2: GOTO 85
80 Y$=Y$+LEFT$(U$,N1-N2):R=N1
85 N=P+R:FOR I=1 TO N :IF VAL(MID$(X$,I,1))<VAL
   (MID$(Y$,I,1)) THEN SWAP X$,Y$:W=1:GOTO 95
90 NEXT
95 CLS:LOCATE 10,4:PRINT X$:LOCATE 10,5:PRINT Y$
100 LOCATE 6,5:PRINT "－":LOCATE 4,7:PRINT "---------- "
105 FOR I=N TO 1 STEP －1
110 A=VAL(MID$(X$,I,1)):B=VAL(MID$(Y$,I,1)):IF A<
    (B+C) THEN Z=A+10－B－C:C=1:GOTO 118
```

```
115 Z=A-B-C:C=0
118 R$(I)=RIGHT$(STR$(Z),1)
120 LOCATE 9+I,8:PRINT R$(I)
125 IF I=P+1 THEN I=I-1:R$(I)=".": GOTO 120
130 IF C=1 THEN LOCATE 8+I,6:PRINT "1"
135 FOR J=0 TO 500:NEXT:NEXT
140 IF R$(N)="0" THEN N=N-1:GOTO 140
145 K=0:PRINT:IF W=1 THEN PRINT X1$"-"Y1$"=-";:
    GOTO 155
150 PRINT X1$"-"Y1$"=";
155 K=K+1:IF R$(k)="0" THEN 155
160 FOR I=K TO N:PRINT R$(I);:NEXT:END
```

程序说明：

(1)105～135 行完成减法运算。

(2)140～155 行对差整形，使输出的结果符合记数习惯，并打印算术横式。

(3)输入整数时，要求在数尾加小数点和一个"0"。如 15，输入 15.0。

(4)本程序未加结束处理，若需要，读者可自己添加。

6.4 程序的设计步骤和方法

有了小数的加、减运算程序 6－2、6－4 后，我们可以着手编制代数加减混合运算程序。可以看出，程序 6－2、6－4 只适用正数的运算，这在小学数学运算中可以满足要求。然而在代数运算中，往往要处理带符号的数据，可能有负数加正数，也可能有负数减正数等。在运算中当符号不同时，要用绝对值大的数减绝对值小的数，然后再根据绝对值大的数的符号决定差的符号。因此，对运算结果，既要考虑差的绝对值的大小，还要考虑差的正负。另外，在实际

运算中，不仅仅是两个数之间相加、减，往往要对一组数进行连续的加减运算，这就要把上一次的运算结果转化为下一轮运算的数据，再与输入的数据进行运算。因而，在设计程序时，要周密考虑，全面安排，以保证程序的通用性和正确性。下面结合这一程序的编写介绍系统的程序设计步骤和方法。

一般来说，程序的编写要经过以下十个步骤：

1. 分析问题性质　　2. 建立数学模型
3. 选取运算方法　　4. 确定设计思路
5. 划分功能模块　　6. 绘制程序框图
7. 分段编写程序　　8. 汇总合成程序
9. 讲究设计技巧　　10. 检查通用合理

下面结合实例，介绍十个步骤的实施过程

例 6.1　编写任意数的高精度加减混合运算程序。要求程序有以下主要功能：

(1)具有演示运算功能。能够列出笔算竖式，演示加、减运算过程。以作为辅助教学程序。

(2)具有连续运算功能。可进行任意个数据的连续加减运算。以适应财会、统计、金融等部门的数字累加工作的需要。

(3)具有重新运算功能。以便对多组数据进行加、减运算。

(4)具有高精度运算功能。以适应用户对运算精度的不同要求。

根据提出问题的要求，按照程序的设计步骤逐条分析讨论如下：

6.4.1　分析问题性质

分析待处理问题的性质是编制程序的基础，也是初学者容易忽视的问题，因此，必须认真对待。分析问题的性质，就是分析待编程序要解决什么问题，有什么特殊要求，变量够不够用，有多大的信息量等等。只有对这些问题进行仔细的分析，才能找到解决问题

的方法。

对于例1，我们要做如下分析：

1. 在带符号数的加、减运算中，第一数(a)、第二数(b)的符号及绝对值的大小，决定了运算结果(c)的符号及绝对值。归纳起来，它们之间分别有八种不同的组合和对应关系。设|a|=A、|b|=B、|c|=C 可列出下表(见表 6－1)：

表 6－1

a 的符号	b 的符号	绝对值比较	运算式	c 的符号
+	+	A>B	A+B=C	+
+	+	A<B	A+B=C	+
+	−	A>B	A−B=C	+
+	−	A<B	B−A=C	−
−	+	A>B	A−B=C	−
−	+	A<B	B−A=C	+
−	−	A>B	A+B=C	−
−	−	A<B	A+B=C	−

由表 6－1 可见，必须针对两数的符号和绝对值，确定运算方式和结果的符号，这是本问题的关键和难点。

2. 必须设计两种运算模块：加法、减法。

3. 在进行加法运算时，要处理好加法和进位，在进行减法运算时，要考虑到减法的借位。

4. 由于是通用加减运算程序，必须以处理小数为主要编写对象，同时也能处理整数的运算。

5. 数据的输入要求符合人们的书写习惯及对电子计算器的使用习惯。即使用微机象使用计算器那样方便。输入正数，不要带“＋”号；输入负数，要输入“－”号；输入整数，不要带任何附加符号等。

6. 为实现多个数据的连续运算，要求程序能自动把上一次的运算结果转化为下一次运算的数据。

7. 进行加、减运算后，运算结果可能会出现前后有零的现象。

如 005.32000，必须把它整形为 5.32 的形式，以符合人们的计数习惯。

8. 为达到高精度的要求，必须放弃数据变量的使用，采用字符串变量装数据。

9. 处理小数运算，必须正确判断小数点的位置，做到两数小数点前后的对应位相加减，不能错位。因此，必须在运算前，对两数进行统一格式处理。

通过上述分析，我们对本题已有了较清楚全面的了解，那么，在编程时就做到了心中有数。

6.4.2 建立数学模型

任何一个实际问题，最后都要用数学表达式来描述，然后再按照 BASIC 规则书写程序。因此，把实际问题抽象成数学模型，是编制程序的又一重要方法。

例如：对 1～100 的自然数求和，其数学表达式为

$$\sum_{n=1}^{100} n$$

又如：车站售票规定，身高 1.1 米以下儿童免票；超过 1.1 米低于 1.3 米买半票；1.3 米以上买全票。

对此问题，可用下述公式表达

$$X=\begin{cases} 0 & H<1.1\text{ 米} \\ 0.5Y & 1.1\leqslant H<1.3 \\ Y & H\geqslant 1.3 \end{cases}$$

式中：Y 为全票价，H 为身高。

由此可见，不同的问题，数学模型是不一样的，而一旦正确建立了数学模型，编制程序就容易多了。对于例一，由于求和就是数学运算，所以这一步可以省去。

6.4.3 选用计算方法

一个计算问题可能在不同的条件下使用不同的公式。而同一个问题，即使有相同的公式也可能有不同的计算方法。选用好的计

算方法，是设计高质量程序的核心。

如求 $\sum_{n=1}^{100} n = 1+2+3+4+\cdots\cdots+100$

下面给出三种解题方法。

1.
```
10 C=0
20 FOR I=1 TO 100
30 C=C+I
40 NEXT
50 PRINT"C=";C:END
```

2.
```
10 C=0:I=0
20 I=I+1:IF I>100 THEN 40
30 C=C+I:GOTO 20
40 PRINT "C=";C:END
```

3. 由于求前 100 项和的通项式为

$\sum_{n=1}^{100} n = (100+1)\times 100 \div 2$　　则有

```
10 PRINT "C=";(100+1) * 100/2
20 END
```

第一种方法使用了循环语句；第 2 种方法使用了转向语句；第 3 种方法使用了计算公式。可见，第 3 种最为简练，执行速度也最快。

总之，解决问题的方法多种多样，我们应尽量选用既节省机时，又节省存储空间的方法。例一是求加、减运算，已是最基本的运算方法，所以本步骤可以省去。

6.4.4 确定设计思路

确定设计思路是编制任何程序都要认真考虑的问题。在弄清问题的性质、特点、要求和确定了运算方法之后，对如何解决这一问题以达到设计要求，应进行周密的思考、反复的推敲，全面的分析。分析得越透彻，考虑的越周到，出错的机会就越少，编程就越顺

利。那么我们要编制的程序应如何确定设计思路呢？可以从以下几个方面来考虑：

1. 为了实现高精度运算，引入串变量 X$、Y$ 装运算数据 X、Y。

2. 为确定运算方式是求差还是求和，设置变量 U、V 记录 X、Y 的符号变化。通过 U、V 的比较，确定运算方式和结果的符号。

3. 为保证两符号不同时，能正确地进行减法运算，要对 X$、Y$ 中装的数据进行绝对值大小的比较，把绝对值大的数据装入 X$ 中，并设置一交换记录变量，以记录是否交换。

4. 为达到辅助教学的目的，必须列出问题横式，运算竖式及结果横式三种对照形式。使学生在运算操作中加深认识、明确概念、掌握代数的加法运算要领，因此要对横式的打印格式进行判断确定。

根据分析问题性质的九个方面和上述四点，确立了我们的设计思路是：输入数据 X、Y 分别装入 X$、Y$ 中；先去除符号；再进行统一格式化；然后判断两数绝对值的大小；绝对值大的数装入 X$；再根据两数符号决定运算方式；最后根据两数符号及绝对值，确定输出横式格式及结果的符号；并把结果转化为 X，装入 X$，准备下一轮计算。

6.4.5 划分功能模块

根据总体需要和系统配置的可能，把程序按功能划分为几个功能模块，将给编程带来许多方便。这样做的好处：一是结构清楚，层次分明；二是功能明确，便于移植；三是可以分头编写，缩短周期。这在大型程序设计中经常用到。对于我们要解决的问题，根据设计思路可如下设置功能模块（见图 6—1）：

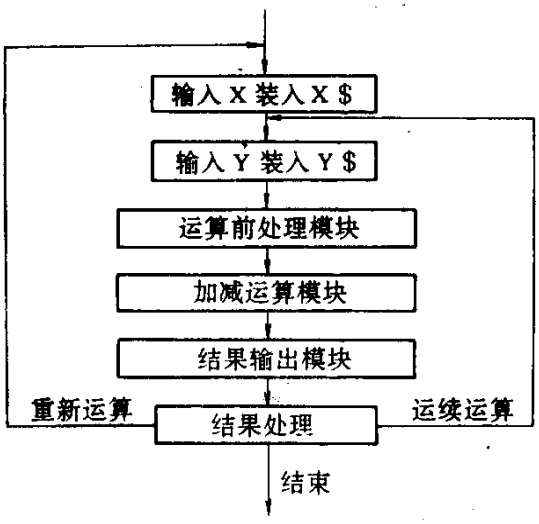


图 6－1　加减运算程序模块结构图

各模块的功能如下：

1. 运算前处理模块的功能是在输入的数据 X、Y 分别装进 X$、Y$ 串变量中以后，对 X、Y 去负号；判别小数点位置；对两数格式化；比较|X|、|Y|的大小；把绝对值大的数装入 X$ 中。

2. 加减运算模块的功能是根据两数的符号，决定加、减运算方式；并完成加、减运算；同时列出运算竖式，模拟运算过程。

3. 结果输出模块的功能是对运算结果整形；确定横式的打印格式和运算结果的正负；输出结果；并把结果转化为 X，装入 X$ 中。

可以看出，这样做使得整个程序阅读、理解、调试变得极为方便，而且也便于修改、交流和移植。

6.4.6　绘制程序框图

根据划分的功能模块绘制程序框图是编制程序的重要手段。

什么叫程序框图呢？我们说，程序框图是一种流程图，它是用各种几何图形及文字说明来直观地描述计算过程的有向图。

程序框图常用的符号有下面一些：

⬭ 椭形框：表示程序的开始或结束。

□矩形框：表示赋值或完成运算等操作。

◇菱形框：判断某个关系式是否成立。如果关系式成立，程序向 Y(Yes)的方向进行；否则向 N(No)的方向进行。

▱斜框：表示输入输出。

→箭头：表示流程的流向。

@ 小园圈内一个字母或数字：表示流程图的接点，有些流程图很长或很复杂，一页纸画不下，需转到下一页时，把一些地方断开，在断开的两头画圈，内标同样的符号，表示这两点是连在一起的。

对于加法程序，我们可以画出如下流程图：

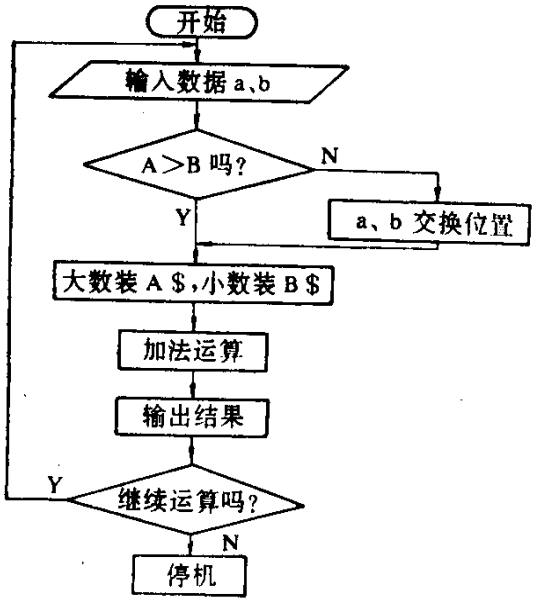


图 6－2　加法程序框图

从上图可以看出，使用程序框图，可以形象地表达程序的设计思路，直观地看清程序的结构。这种形式，便于阅读、检查、修改程序。学会并习惯画框图是编程的基本功。它可以使程序减少错误，便于交流。不难想象，设计一个大型应用程序，不使用框图是很难顺利完成的。

下面画出解决我们主题(例 1)的程序框图如图 6－3。

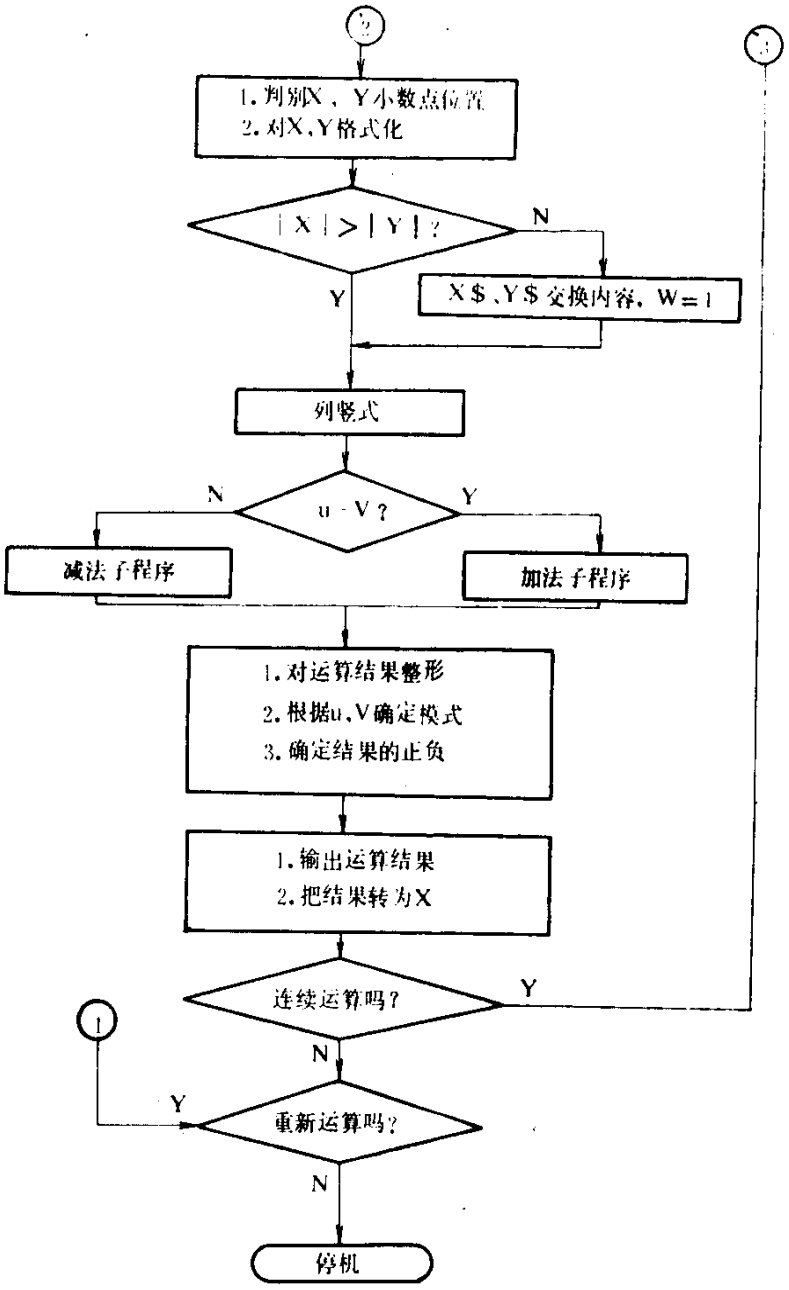


图 6－3　加减运算程序框图

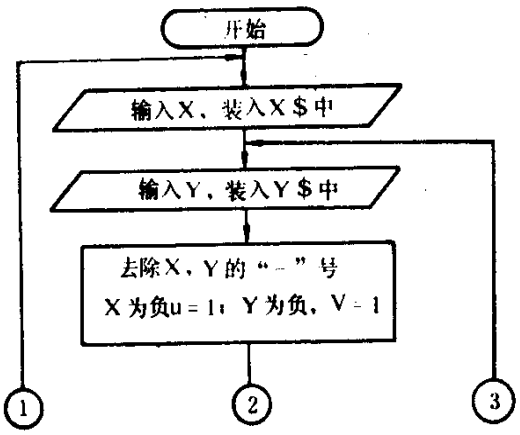


图 6－3　加减运算程序框图

框图中引入了 U、V 两个变量，记录两数的正负。为正时，其值为 0，否则其值为 1。通过比较 U、V 的值，可以方便地确定进行的运算是求差还是求和。

6.4.7　分段编写程序

分段编写程序就是根据程序框图的要求，编写出相应的处理程序。

在程序 6－2、6－4 中，我们已介绍了数据装入、列竖式演示、两数格式化及加、减运算程序，在此不再赘述。下面重点介绍两数大小的比较、算术横式格式的确定和运算结果转化程序的编写。

关于数学横式的格式，根据附表所列的八种组合，共有四种形式。分别为：

1.　X＋|Y|＝|Z|（X、Y 都为正数或者|X|＜|Y|而 X 为负，Y 为正时）

2.　X－|Y|＝－|Z|（X、Y 都为负时或者|X|＜|Y|，X 为正，Y 为负时）

3.　X＋|Y|＝－|Z|（|X|＞|Y|且 X 为负，Y 为正时）

4.　X－|Y|＝|Z|（|X|＞|Y|，且 X 为正，Y 为负时）

相应的处理程序如下：（通过 U、V、W 三个变量进行判别。）

```
150 IF (U=0 AND V=0) OR (U=1 AND V=0 AND W=1) THEN
    PRINT X1$"+"Y1$"=";
155 IF(U=1 AND V=1)OR(U=0 AND V=1 AND W=1) THEN X
    $="-":PRINT X1$;Y1$"=-";
160 IF U=1 AND V=0 AND W=0 THEN X$="-":PRINT X1
    $"+"Y1$"=-";
165 IF U=0 AND V=1 AND W=0 THEN PRINT X1$;Y1$"=";
```

150～165 四行分别对应了 1～4 四种横式。

关于运算结果的转化可这样解决。设运算结束后，运算结果已存于 R$(X)数组中，最高位为 R$(K)，最低位为 R$(N)，编程如下：

```
135 X$="0"
170 FOR I=K TO N:PRINT R$(I);:X$=X$+R$(I):R$(I)=
    "":NEXT
175 X=LEN(X$):IF LEFT$(X$,1)="0" THEN X$=RIGHT
    $(X$,X-1)
```

175 行的作用是当结果为正数时，除去数据最前面的"0"。

关于两数大小的比较可编制如下程序：(设两数已统一格式化)

```
350 FOR I=1 TO N:AA=VAL(MID$(X$,I,1)):BB=VAL(MID
    $(Y$,I,1)):IF AA>BB THEN RETURN
355 IF AA<BB THEN SWAP X$,Y$:W=1:RETURN
360 NEXT:IF U=V THEN RETURN
365 PRINT "JIE Guo='0'":X$="0":GOTO 10(转连续运算入口)
```

当两数绝对值相同，符号不同时，其和为零。所以设置 365 行作为结果为"0"时的处理。

6.4.8 汇总合成程序

通过以上讨论，参考程序 6－2、6－4，我们可以方便地编出例一的程序(见程序 6－5)。

程序 6－5：任意数加减混合高精度运算程序

```
1 REM"N0. 6－5 sup＋－"
2 U$="                                                    "
5 CLS:CLEAR:DIM R$(255):INPUT"X=";X$
10 INPUT "Y=";Y$:X1$=X$:Y1$=Y$:N1=N2=C=P=R
   =U=V=W:X1=LEN(X$):Y1=LEN(Y$)
15 IF LEFT$(X$,1)="－" THEN U=1:X1=X1－1:X$=RIGHT
   $(X$,X1)
20 IF LEFT$(Y$,1)="－" THEN V=1:Y1=Y1－1:Y$=RIGHT
   $(Y$,1)
25 N1=N1＋1:IF N1=X1＋1 THEN X$=X$＋".0":GOTO 35
30 IF MID$(X$,N1,1)<>"." THEN 25
35 N2=N2＋1:IF N2=Y1＋1 THEN Y$=Y$＋".0":GOTO 45
40 IF MID$(Y$,N2,1)<>"." THEN 35
45 IF N2<N1 THEN 60
50 IF N2=N1 THEN P=N1:GOTO 65
55 X$=LEFT$(U$,N2－N1)＋X$:P=N2:GOTO 65
60 Y$=LEFT$(U$,N1－N2)＋Y$:P=N1
65 N1=LEN(X$)－P:N2=LEN(Y$)－P
70 IF N2<N1 THEN 85
75 IF N1=N2 THEN R=N1:GOTO 90
80 X$=X$＋LEFT$(U$,N2－N1):R=N2:GOTO 90
85 Y$=Y$＋LEFT$(u$,N1－N2):R=N1
90 N=P＋R:CLS:IF V=1 THEN PRINT X1$;Y1$"=?":GOTO
   100
95 PRINT X1$"＋"Y1$"=?"
100 GOSUB 350
105 LOCATE 10,4:PRINT X$:LOCATE 10,5:PRINT Y$
110 LOCATE 4,7:PRINT"－－－－－－－－－－－－－－－－－－－－
    －－－－－－－－"
115 IF U=V THEN LOCATE 6,5:PRINT "＋": GOSUB 200
```

```
120 IF u<>v THEN LOCATE 6,5:PRINT"-":GOSUB 300
125 IF R$(N)="." THEN N=N-1:GOTO 135
130 IF R$(N)="0" THEN N=N-1:GOTO 125
135 K=-1:X$="0":PRINT:PRINT
140 K=K+1:IF R$(K)="0" OR R$(K)="" THEN 140
145 IF R$(K)="." THEN R$(K)="0."
150 IF (U=0 AND V=0)OR (U=1 AND V=0 AND W=1) THEN
    PRINT X1$"+"Y1$"=";
155 IF (U=1 AND V=1) OR (U=0 AND V=1 AND W=1) THEN
    X$="-":PRINT X1$;Y1$"=-";
160 IF U=1 AND V=0 AND W=0 THEN X$="-":PRINT X1
    $"+"Y1$"=-";
165 IF U=0 AND V=1 AND W=0 THEN PRINT X1$;Y1$"=";
170 FOR I=K TO N:PRINT R$(I);:X$=X$+R$(I):R$(I)=
    "":NEXT
175 X=LEN(X$):IF LEFT$(X$,1)="0" THEN X$=RIGHT
    $(X$,X-1)
180 LOCATE 10,20:PRINT "LIAN XU JI SUAN MA?(Y/N)";"Q$
    =INKEY$(0):IF Q$<>"N" THEN PRINT:PRINT "X=" X
    $:GOTO 10
185 LOCATE 10,22:PRINT "CHONG XIN JI SUAN MA? (Y/N)";:
    Q$=INKEY$(0): IF Q$<>"N" THEN 5
190 CLS:LOCATE 10,10:PRINT"GOODBYE!!!":END
200 FOR I=N TO 1 STEP-1
205 A=VAL(MID$(X$,I,1)):B=VAL(MID$(Y$,I,1)):Z=A
    +B+C:C=0
210 IF Z>9 THEN C=1
215 R$(I)=RIGHT$(STR$(Z),1)
220 LOCATE 9+I,8:PRINT R$(I)
225 IF I=P+1 THEN I=I-1:R$(I)=".": GOTO 220
230 IF C=1 THEN LOCATE 8+I,6:PRINT "1"
```

```
235 FOR J=0 TO 500:NEXT:NEXT
240 IF C=1 THEN R$(0)="1":LOCATE 9,8:PRINT "1"
245 RETURN
300 FOR I=N TO 1 STEP-1
305 A=VAL(MID$(X$,I,1)):B=VAL(MID$(Y$,I,1)):IF A<
    B+C THEN Z=A+10-B-C:C=1:GOTO 315
310 Z=A-B-C:C=0
315 R$(I)=RIGHT$(STR$(Z),1)
320 LOCATE 9+I,8:PRINT R$(I)
325 IF I=P+1 THEN I=I-1:R$(I)=".":GOTO 320
330 IF C=1 THEN LOCATE 8+I,6:PRINT"1"
335 FOR J=0 TO 500:NEXT:NEXT
340 RETURN
350 FOR I=1 TO N:AA=VAL(MID$(X$,I,1)):BB=VAL(MID
    $(Y$,I,1)):IF AA>BB THEN RETURN
355 IF AA<BB THEN SWAP X$,Y$:W=1:RETURN
360 NEXT:IF u=v THEN RETURN
365 PRINT "JIE GUO='0'":X$="0":GOTO 180
```

程序说明：

1. 25～35 行是对输入的整数，自动补小数点和一个"0"，以便格式化。

2. 45～85 行完成两数的格式化。

3. 90、95 两行是打印问题的横式。

4. 105～110 行为列竖式语句。

5. 125～145 行的作用是对运算结果整形。

125 行对整数结果去小数点。130 行去除结果小数后的"0"。140 行去除结果整数部分前面的"0"。145 行对无整数部分结果的前面添"0"。125、145 行的目的是使结果符合人们的记数习惯。

6. 200～245 行为加法子程序。

7. 300～340 行为减法子程序。

8. 350～365 行为比较|X|、|Y|大小的子程序。

键入程序。运行后屏幕显示

X＝?　　　　　　　　键入－112.53↙

Y＝?　　　　　　　　键入 3264.2↙

清屏。显示

－112.53＋3264.2＝?

```
   3264.2
    112.53
－     1 1
───────────
   3151.67
```

－112.53＋3264.2＝3151.67

LIAN XU JI SUAN MA? (Y/N)(询问连续计算吗?)

若连续计算，则键入除"N"键外的任一键。则显示

X＝3151.67

Y＝?　　　　　　　　键入－6127.653↙

清屏。显示

3151.67－6127.653＝?

```
   6127.653
   3151.67
－  11 1 1
────────────
   2975.983
```

3151.67－6127.653＝－2975.983

LIAN XU JI SUAN MA? (Y/N)

若键入"N"则显示

CHONG XIN JI SUAN MA? (Y/N)(询问重新计算吗?)

若重新运算则键入除"N"键外的任一键。

若结束运算则键入"N"键。

清屏。显示

GOODBYE!!!

停机。

程序使用说明：

1. 本程序可对任意整数、小数、带符号数进行 30 位以内的加减运算。各种数字的输入可按正常方式(如使用计算器那样)操作，没有特殊要求。

2. 由于要在屏幕上列竖式演示，受屏幕所限，输入数据的位数限于 23 位以下。若要进行更多位数据的计算，可删去列竖式语句。

3. 程序清单是按演示运算给出的(最完全的程序)。若用于会、统计算，可删去列竖式语句及 235、335 行中的延时语句，并将 180 行改为 GOTO 10 则可方便迅速地完成任意个数据的连续加减。

6.4.9 讲究设计技巧

设计程序入门并不难，但要编制高质量的程序却并非易事。编好一个程序，仅凭了解计算机语言的基本语句是很不够的。程序设计是一项技术性较强的工作，它是模仿性和创造性的统一；同时它又是一项极其复杂而又十分灵活的工作，需要人们进行不断探索、不断创造。

作为本文的中心议题，以后各章节将列举较多的示例介绍程序设计技巧的各个方面。例如在程序 6—5 中，关于 X、Y 两数绝对值大小的比较，我们是安排在格式化以后进行的。仅使用了四行语句就较好地完成了这一工作。若安排在其它位置进行大小的比较，虽然也可以，但程序要复杂得多。例如在输入数据后立即进行大小的判断，程序将变得很庞大。为便于对照，特列出格式化前比较两数大小的程序。

```
25 N1=N1+1:IF N1=X1+1 THEN X$=X$+".0":GOTO 35
30 IF MID$(X$,N1,1)<>"." THEN 25
35 N2=N2+1:IF N2=Y1+1 THEN Y$=Y$+".0":GOTO 45
40 IF MID$(Y$,N2,1)<>"." THEN 35
45 A=LEN(X$):B=LEN(Y$)
```

```
50 IF N1>N2 THEN 后续程序(|X|>|Y|)
55 IF N1<N2 THEN SWAP X$,Y$:GOTO 后续程序(|X|<|Y|,
   交换X$,Y$的内容)
60 FOR I=1 TO N1:AA=VAL(MID$(X$,I,1)):BB=VAL(MID
   $(Y$,I,1))
65 IF AA>BB THEN 后续程序(|X|>|Y|)
70 IF AA<BB THEN SWAP X$,Y$:GOTO 后续程序
75 NEXT
80 IF A-N1>B-N2 THEN 110
85 IF A-N1<B-N2 THEN 130
90 FOR I=N1+1 TO A:AA=VAL(MID$(X$,I,1)):BB=VAL
   (MID$(Y$,I,1))
95 IF AA>BB THEN 后续程序(|X|>|Y|)
100 IF AA<BB THEN SWAP X$,Y$:GOTO 后续程序
105 NEXT:IF U<>V THEN PRINT "JIE GUO='0'":GOTO 结束
    处理程序
108 GOTO 后续程序
110 FOR I=N2+1 TO B:AA=VAL(MID$(X$,I,1)):BB=VAL
    (MID$(Y$,I,1))
115 IF AA>BB THEN 后续程序(|X|>|Y|)
120 IF AA<BB THEN SWAP X$,Y$:GOTO 后续程序
125 NEXT:GOTO 后续程序
130 FOR I=N1+1 TO A:AA=VAL(MID$(X$,I,1)):BB=VAL
    (MID$(Y$,I,1))
135 IF AA>BB THEN 后续程序(|X|>|Y|)
140 IF AA<BB THEN SWAP X$,Y$:GOTO 后续程序
145 NEXT:SWAP X$,Y$
150 后续程序
```

可以看出这一比较过程相当繁琐。首先对两数整数部分判断比较，50行判断X的整数部分位数多于Y的情况，这时|X|>|Y

|；55 行判断 X 的整数部分位数少于 Y 的情况，这时|X|<|Y|；60～75 行判断当 X、Y 的整数部分位数相同时|X|、|Y|的大小，其中也考虑了三种情况|X|>|Y|；|X|<|Y|；|X|=|Y|。其次当两数整数部分相等时，要对两数小数部分判断比较。这也要考虑三种情况，即 X 的小数部分的位数多于 Y、等于 Y 和小于 Y。而对每种情况还要进行三次比较，即|X|>|Y|；|X|<|Y|；|X|=|Y|。它们分别由 90～105、110～125；130～145 三段程序实现。

上述程序的正确性是不容置疑的，但同程序 6－5 相比上述程序显然不合理。尽管可把循环比较程序设计为子程序调用，可使程序缩短，但肯定还会比程序 6－5 的 350～365 语句复杂得多。由此可见，某种操作安排的前后顺序将极大地影响程序的合理性。因此编程中要讲究设计技巧，对每段程序反复推敲，以求设计出最简程序。

6.4.10 检查通用合理

为保证程序的通用性和逻辑上的合理性，一个程序编写完毕必须做如下几方面的检查：

1. 检查是否安排了判断输入数据是否正确的语句。如在编制十进制数学运算程序时要考虑输入字母的处理；在编制除法程序时要考虑除数为“0”的处理。

2. 对使用某一程序段来完成重复计算的工作应检查是否能正确的调用和返回。

3. 对多次重新运行的程序，应保证任何情况下都正确无误。如程序 6－5 在连续运算时，在 10 行对有关变量清零，若没有该语句，连续运算将出现错误。

4. 有时程序中应考虑可能出错的自动处理。如在程序 6－5 中 365 行对结果为零时的拦截处理等。

通过详细的检查，保证程序在各种情况下运行正常，计算正确。

以上我们从十个方面介绍了编程的一般步骤和方法。实际工作中也不一定完全按步就班的进行。如问题的开始就给出了计算公式，则建立数学模型一步就可省去；若问题比较简单则划分功能模块和绘制程序框图工作也可以不做。总之要审时度势、灵活处理。这些在以后的讨论中和编程实践中将逐步介绍。

6.5 分数的加减运算

与整数、小数的加减运算相比，分数的加、减运算要复杂得多。因此目前多数微机都不能直接进行分数的运算处理。而且分数的加减运算又比乘除运算复杂。分数在加减运算前要先通分，变为同分母后才能进行分子相加减，分子运算结束后，还要对和或差的分子、分母进行约分，以求输出结果为最简分数。因此必须采取有效措施进行分数通分、约分的特殊操作。下面先介绍如何求得两数的最小公倍数 LCM 和最大公约数 GCM。

6.5.1 最小公倍数和最大公约数

我们知道 LCM(Least Common Multiple)和 GCM(Greatest Common Measure)是以如下方式进行的。如求 18、45 的 GCM 和 LCM，首先从 2 开始依次增大逐数判断，是否能被两数同时整除。见下式：

```
3 | 18   45
3 |  6   15
  |  2    5
```

第一次判断 3 为第一公约数，能同时被两数整除，则把 3 写在左边，18、45 同时除以 3，商写在 18、45 的下面。第二次再对 6 和 15 从 2 开始依次增大，逐数判断，得出第 2 公约数仍为 3。把第二公约数写在左边，6 和 15 同时除以 3，把商 2、5 分别写在 6、15 的下面。再比较 2 和 5，现在除 1 之外已没有公约数了。则计算最大公

约数＝所有公约数之积。

即　GCM＝3×3＝9

最小公倍数 LCM＝3×3×2×5＝90

或　LCM ＝48×18÷GCM

＝45×18÷9＝90

根据笔算过程，我们可直接编写出通分、约分子程序。（见子程序 6.10）

子程序 6.10：通分、约分程序(A)

```
10 CLS:CLEAR:INPUT "A,B=";A,B
20 IF A<B THEN SWAP A,B
30 C=1
40 FOR I=2 TO B
50 IF A=A/I*I AND B=B/I*I THEN 80
60 NEXT:X=A*B*C
70 PRINT "LCM="X, "GCM="C:END
80 A=A/I:B=B/I:C=C*I:GOTO 40
```

子程序 6.10 的求解过程是模仿笔算方式进行的。

程序说明：

1. 20 行判断 A＜B 时(如 18＜45)则令 A、B 交换，以保证变量 A 中始终是较大的数。

2. 40 行置循环语句。循环终值以较小的数为确定依据。

3. 50 行判断 A、B 两数能否被 I 整除，不能整除，I＋1 再试。若能整除则转 80 行，记录第一公约数，并使 A、B 同时除以第一公约数。然后再进行第二轮循环判断，直到循环结束，输出结果。

4. 60 行计算最小公倍数。也可以改写为 X＝X/C，但必须在 30 行中加入 X＝A＊B。

关于求最小公倍数和最大公约数，还有更简便的方法。（见子程序 6.11）

子程序 6.11：通分、约分程序(B)

```
10 CLS:CLEAR:INPUT "A,B=";A,B:X=A*B
20 IF A<B THEN SWAP A,B
30 A=A-A/B*B:IF A=0 THEN 50
40 GOTO 20
50 X=X/B:PRINT "LCM=";X,"GCM=";B:END
```

子程序 6.11 是根据欧几里德辗转相除原理设计的，在此不多介绍。有兴趣的读者可参阅华罗庚编著的《数论导引》一书。

6.5.2 编写程序

解决了最小公倍数和最大公约数的运算问题，就可以编制分数加减运算程序了。(由于程序比较简单，采取直接编写方法。)编程思路是：首先输入参与运算的分数个数，定义两个数组分别装分子、分母。第二、取出两个分数进行加减运算，运算前先通分，运算后再约分，求出最简分数后，再与第三个分数相加减。最后输出运算结果。程序如下：

程序 6－6：分数的加减运算程序

```
5 REM "NO. 6-6 FEN SHU+-"
10 CLS:CLEAR:INPUT "JiSuan Fen Shu Ge Shu:N=";N:DIM FZ
   (N),FM(N)
20 FOR I=1 TO N:PRINT "Shu Ru Di"I"Ge Fen Shu:";:INPUT F
   $:L=LEN(F$)
30 FOR J=1 TO L:Z$=MID$(F$,J,1):IF Z$="/" THEN 50
40 NEXT
50 FZ(I)=VAL(LEFT$(F$,J-1)):FM(I)=VAL(RIGHT$(F$,
   L-J)):NEXT
60 FZ=FZ(1):FM=FM(1)
70 FOR I=2 TO N:A=FM:B=FM(I):GOSUB 150
80 AA=D/FM*FZ+D/FM(I)*FZ(I):BB=D:A=AA:B=BB:GO-
   SUB 160
90 FZ=AA/C:FM=BB/C:NEXT
100 PRINT FZ(1)"/" FM(1);
```

```
110 FOR I=2 TO N:IF FZ(I)<0 THEN 130
120 PRINT "+" FZ(I)"/"FM(I);:GOTO 140
130 PRINT FZ(I)"/"FM(I);
140 NEXT:PRINT "="FZ"/" FM:END
150 X=A*B
160 IF A<B THEN SWAP A,B
170 A=A-A/B*B:IF A=0 THEN 190
180 GOTO 160
190 C=B:C=X/B:RETURN
```

程序说明：

1. 10～50 行把参与运算的所有分数的分子装进 FZ(N)数组，所有分母装进 FM(N)数组。

2. 60～90 行的功能是通分、加减、约分。80 行摩仿笔算过程，即用最小公倍数 D 除以分母得出通分扩大倍数，乘以分子，使分子分母同时增大相同倍数。两分数通分完毕，两分子相加，分子之和赋值给 A，分母赋值给 B，转 150 行求出最大公约数，转 90 行进行约分，变为最简分数后，再与下一个分数相加减。

3. 100～140 行是结果输出打印程序。

4. 150～190 行是求最小公倍数，最大公约数子程序。

该程序的使用方法如下：

程序运行后，根据提示首先输入参与运算的分数个数，再依次输入各分数，每输入一个分数回车一次。

如：有如下算式

$$\frac{1}{2}-\frac{1}{3}+\frac{3}{5}-\frac{5}{7}$$

输入过程如下：

```
4 ↙
1/2 ↙
-1/3 ↙
3/5 ↙
-5/7 ↙
```

通过阅读程序可发现程序 6－6 的处理数据范围仍受整数 BASIC 的限制，分子、分母结果的值须控制在－32768～＋32767 之间。若编制通用分数加减运算程序，则须采用小数加法的方法将分子、分母的每一位分别装入数组中，再进行运算。限于篇幅，不再举例。读者只要依据以上介绍的方法和思路不难编出程序。

6.6 乘法运算

6.6.1 整数乘法

乘法运算程序的数据装入、结果输出和结果处理与加法程序的区别不大，关键在运算部分。下面先看一个笔算三位数乘法的运算过程。如 987×654＝?

一般列竖式计算如下：

```
   987
 ×  654
------
  3948
 49350
592200
------
645498
```

运算过程共四步：

1. 用乘数的个位 4 分别乘以被乘数的各位得出个位乘积 3948。

2. 用乘数的十位 50 分别乘以被乘数的各位得出十位乘积 49350。

3. 用乘数的百位 600 分别乘以被乘数的各位得出百位的乘积 592200。

4. 将三次乘积求和，得出乘积 645498。

根据笔算过程的总结，可以确定我们的设计思路为：

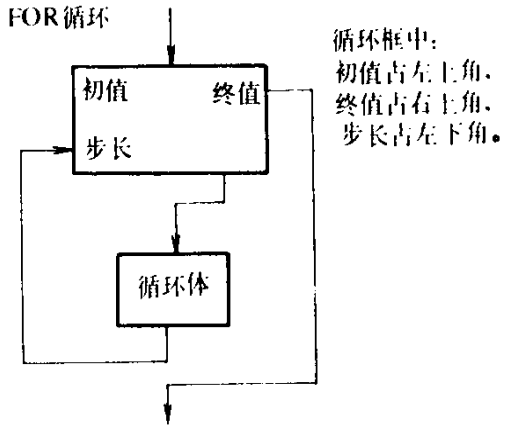
1. 定义一个装乘积的数组，把第一次乘积依次装入乘积的数组中。

2. 把高一位的乘积依次与前一次乘积相加，所得之和再存入乘积数组的对应单元中。

3. 乘数有多少位则进行多少次第二项操作。

4. 与笔算的不同点在于，不是最后求和，而是进行一位求积运算，进行一次求和。

根据设计思路可以画出程序框图。在这里补充介绍循环语句的框图画法。见下图符号：



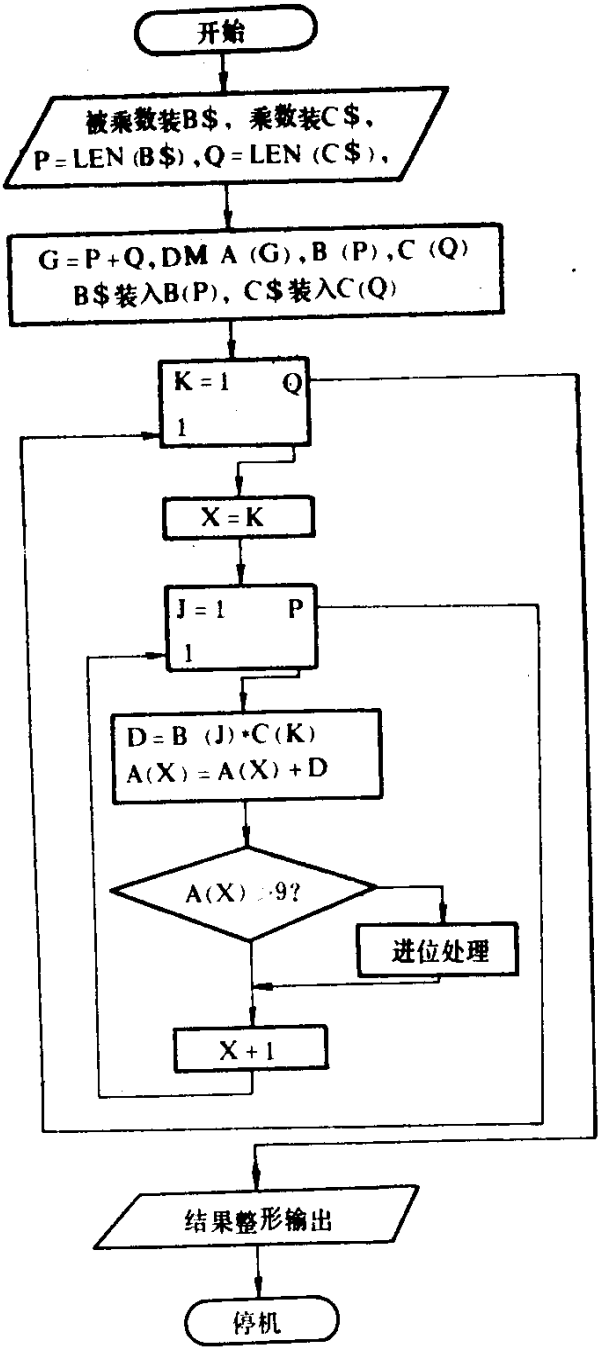


图 6—4　乘法运算程序框图

对照框图，编程如下：

程序 6－7：整数乘法程序

```
1 REM "NO. 6-7 CHENG FA"
5 CLS:CLEAR:INPUT "Bei Cheng Shu B=";B$:P=LEN(B$)
10 INPUT "Cheng Shu C=";C$:Q=LEN(C$)
15 G=P+Q:DIM A(G),B(P),C(Q)
20 FOR I=1 TO P:B(P-I+1)=VAL(MID$(B$,I,1)):NEXT
25 FOR J=1 TO Q:C(Q-J+1)=VAL(MID$(C$,J,1)):NEXT
30 FOR K=1 TO Q:X=K
35 FOR J=1 TO P:D=B(J)*C(K):A(X)=A(X)+D
40 IF A(X)>9 THEN A(X+1)=A(X+1)+A(X)/10:A(X)=A(X)
   -A(X)/10*10
45 IF A(X+1)>9 THEN A(X+2)=A(X+2)+1:A(X+1)=A(X+
   1)-A(X+1)/10*10
50 X=X+1:NEXT
55 NEXT:PRINT:PRINT
60 IF A(G)=0 THEN G=G-1
65 FOR I=G TO 1 STEP-1:PRINT A(I);:NEXT:END
```

程序说明：

1. 30～55 行是乘法运算程序。本程序采取两重循环，外层循环控制乘数各位，内层循环控制被乘数各位。通过外层循环，先取乘数个位，进入内循环后依次与被乘数相乘，相乘之积存入 A(G) 数组中。

2. 35～50 行完成一位乘数与被乘数各位相乘的操作。

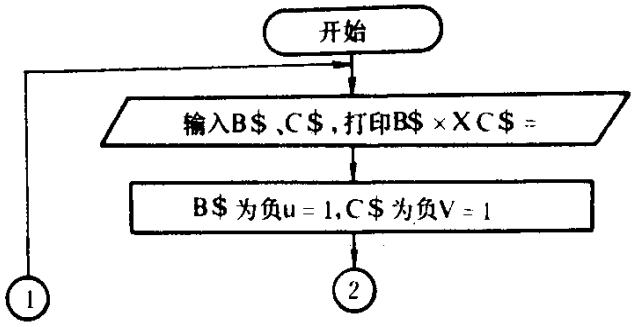
3. 40～45 行完成进位操作。由于两个单位数相乘之积 A(X) 中最大的数为 81，因此进位最大为 8，而 A(X+1)位由于与 A(X) 的进位相加有可能大于 9，故还须考虑向 A(X+2)位的进位问题。

乘法程序的编程技巧关键在于进位处理，只要理解掌握了 40～45 行就掌握了乘法的编程技巧。

6.6.2 小数乘法

上面介绍了整数乘法程序。那么如何编制小数乘法程序呢?

实际上编制小数乘法程序要比小数加法程序简单。因为在乘法运算中是不计小数点的。大家都知道,我们在笔算小数乘法时根本不考虑小数点的存在,按整数进行运算,只在求出乘积后,计算两数小数部分之和记为 N,对乘积由右向左查 N 位点上小数点,就得出两小数的乘积。由此可见,只要去掉小数点就可直接使用程序 6—7 进行求积运算,最后在乘积的适当位置添上小数点就行了。编程的关键是在运算前去小数点;运算后添小数点。同时还要考虑无整数部分小数,在去小数点后,数左面"0"的处理。根据以上分析,可画出程序框图。



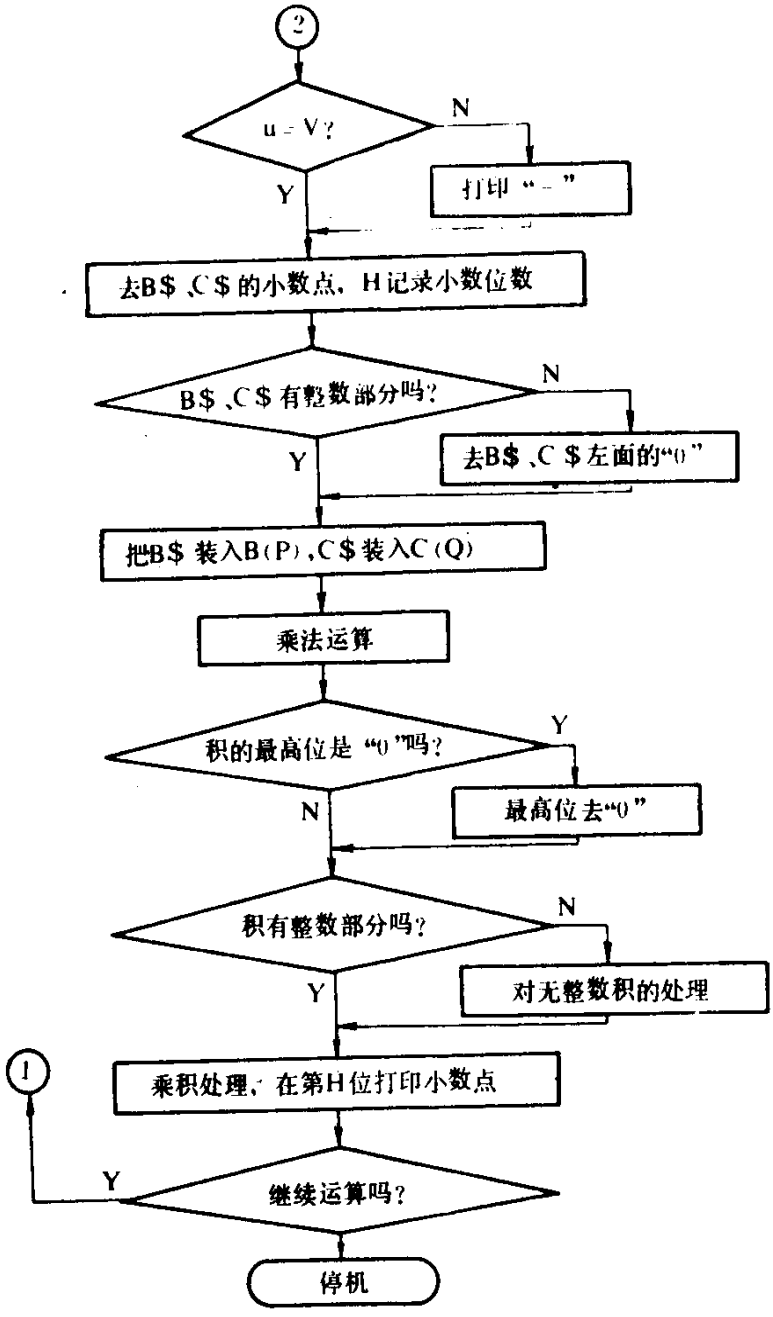


图 6－5　小数乘法运算程序框图

对照程序框图，可编出通用乘法程序

程序 6－8：通用乘法运算程序

```
1 REM "NO. 6－8 supx"
5 CLS:CLEAR:INPUT "Bei Cheng Shu=";B$:B=LEN(B$)
10 INPUT "Cheng Shu=";C$:C=LEN(C$)
15 E=0:F=0:H=0:PRINT B$"X"C$"=";:U=V
16 IF LEFT$(B$,1)="－" THEN U=1:B=B－1:B$=RIGHT
   $(B$,B)
17 IF LEFT$(C$,1)="－" THEN V=1:C=C－1:C$=RIGHT
   $(C$,C)
20 IF U<>V THEN PRINT "－";
25 FOR I=1 TO B:IF MID$(B$,I,1)="." THEN E=I:GOTO 35
30 NEXT:GOTO 40
35 B$=LEFT$(B$,E－1)+RIGHT$(B$,B－E):H=B－E:B=
   B－1
40 FOR J=1 TO C:IF MID$(C$,J,1)="." THEN F=J:GOTO 50
45 NEXT:GOTO 55
50 C$=LEFT$(C$,F－1)+RIGHT$(C$,C－F):H=H+C－F:
   C=C－1
55 DIM A(B+C)
60 FOR I=1 TO B:IF MID$(B$,I,1)="0" THEN NEXT
65 B=B－I+1:B$=RIGHT$(B$,B)
70 FOR J=1 TO C:IF MID$(C$,I,1)="0" THEN NEXT
75 C=C－J+1:C$=RIGHT$(C$,C)
80 P=B:Q=C:G=P+Q
85 DIM B(P),C(Q)
90 FOR I=1 TO P:B(P－I+1)=VAL(MID$(B$,I,1)):NEXT
95 FOR J=1 TO Q:C(Q－J+1)=VAL(MID$(C$,J,1)):NEXT
100 FOR K=1 TO Q:X=K
105 FOR J=1 TO P:D=B(J)*C(K):A(X)=A(X)+D
```

```
110 IF A(X)>9 THEN A(X+1)=A(X+1)+A(X)/10:A(X)=A
    (X)-A(X)/10*10
115 IF A(X+1)>9 THEN A(X+2)=A(X+2)+1:A(X+1)=A(X
    +1)-A(X+1)/10*10
120 X=X+1:NEXT
125 NEXT
130 IF A(G)=0 THEN G=G-1
135 IF G<H THEN G=H:PRINT"0";
145 FOR I=G TO 1 STEP-1
150 IF I=H THEN PRINT".";
155 PRINT A(I);:NEXT:PRINT
160 PRINT "Ji Xu Ma? (Y/N)";:Y$=INKEY$(0):IF Y$<>"N"
    THEN 5
165 CLS:LOCATE 10,10:PRINT "GOODBYE!":END
```

程序说明：

1. 16～20 行的功能是去除符号；判别积的符号。两数符号相同，积为正，否则为负。

2. 25～50 行的作用是去小数点。25 行测试被乘数小数点的位置，若被乘数是小数，则由 E 记录小数点的位置，转入 35 行去掉小数点，由 H 记录小数部分长度，并调整 B$ 的字符长度 B。若被乘数为整数则跳转 40 行判断乘数是否为小数，若乘数是小数则由 F 记录小数点的位置，转入 50 行去小数点，把乘数的小数部分长度增加给小数长度变量 H，并调整 C$ 的字符长度 C。若乘数为整数则跳转 55 行定义装积数组。

3. 60～75 行为整形程序。整数部分为"0"的小数，在去小数点后，会出现数字的高位是"0"的情况。如 0.00038×0.00059，去小数点后分别为 000038、000059。对这种形式的数，虽然能够运算，但有效数以后的运算都是无效循环，可以算出无效循环次数为 32 次，是有效循环的 8 倍；另外，还多占用内存 200 个单元，是有效占

用的5倍。这无疑多占用了机时、内存。因此必须把前面的"0"除去，变为38、59的形式。60行对B$从第一位到第B位逐位判断，若第一位为"0"，再判断第二位，依次进行，直到判断第I位不为零。如000038的第五位不为"0"，则执行65行，令B$的字符长度为B=6-5+1=2，则令B$="38"，舍去前四个"0"。同样，70、75行对C$整形。

4. 80～125行为数据装入数组及运算程序。与程序6-5中的15～55行大同小异，不再赘述。

5. 135～140行的功能是对无整数部分的积进行添"0"处理。如0.00038×0.00059，去小数点整形后为38×59=2242。积的小数部分长度为10，则需在2242前添6个"0"，再加小数点。135行的作用是修正装积数组中应取下标变量的个数，从而使执行155语句时，打印出6个"0"。140行的作用是打印小数点前面的"0"。

6. 145～155行为乘积的输出程序。150行为打印小数点语句。

7. 本程序编写技巧的关键在于小数部分长度H的判断处理。此处较难理解，特着重分析如下：首先在15行中对H置初值，H=0。当被乘数为整数时，即25行中I从1变到B没有读到小数点时，则转40行测试乘数。由于被乘数是整数，H=0。如果被乘数有小数则转35行，令H=B-E，这时H记录了被乘数小数的长度。然后再判断乘数，若乘数为整数则H=B-E，转入运算程序。若乘数为小数则转45行，将乘数的小数长度C-F加给H，使H=B-E+C-F。这时H为两数小数部分长度之和。

8. 程序6-8没有考虑乘数为"0"的处理，若需要，可在20行后加一处理语句。

程序6-8为通用乘法运算程序，它可以精确求解30位以内任意整数、小数的求积运算。若修改数据装入数组语句，则可以求解255位以内，甚至更多位数的求积运算。

关于分数的乘法运算，由于是分子、分母同时相乘，然后再对

乘积进行约分，这只要套用分数加法中的约分程序和整数乘法程序就可解决。留给读者练习。

好了，有了程序 6－8 我们可以做一些有趣的计算，这些计算结果有些即使在大型微机上也难以看到。数学中把左右对称的数叫回文数，如 363、26762、1991 等都叫做回文数。自然数中有无限个回文数，你见过下面的回文数吗？

① 12345678987654321

② 121242363484363242121

③ 1210242036304840363024201121

④ 123210246420369630246420123321

⑤ 10203040506070809080706050403020 1

⑥ 1002003004005006007008009008007006005004003002001

不要以为这些数是杜撰的，请输入下列算式试一试：$(111111111)^2$、$(11011011011)^2$、$(11001100110011)^2$、$(111000111000111)^2$、$(1010101010101010)^2$、$(1001001001001001001001001)^2$。

6.7 简单整数除法

文章的开始已讲过整数 BASIC 语言的除法运算精度较差，商没有小数部分，实用价值不大。为了提高整数 BASIC 除法运算能力，精确求商，我们可以模仿人们笔算除法的过程，先求商的整数部分，把它打印出来，把余数扩大 10 倍（即笔算除法中的不够除补“0”），用余数去除以除数，求得小数点后第一位，再把余数扩大 10 倍，再用余数去除以除数，求得小数点后第二位，如此循环往复，只要商不是有限小数就可以一直进行下去，直到所希望的精度为止。

我们先讨论简单整数除法的高精度运算程序的编写。根据以上分析，可画出程序框图如下（见图 6－6）：

根据程序框图我们可分段编写出相应的处理程序。

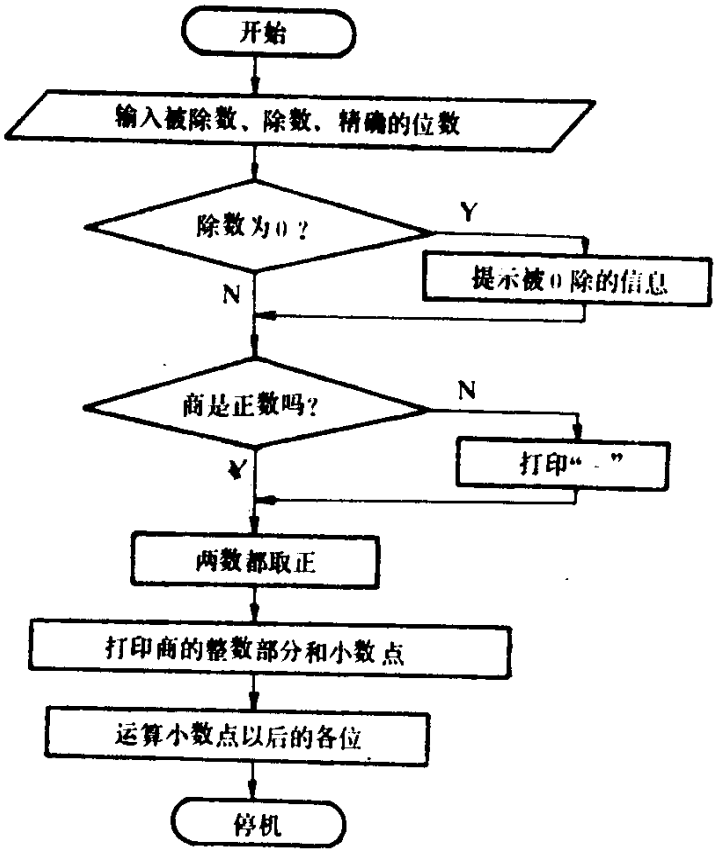


图 6－6　简单整数除法运算程序框图

若以 X 表示被除数，Y 表示除数，N 表示精度（即小数点后的位数），以 M 为商，M＝X/Y，然后安排一个循环，以处理小数点以后各位的运算。循环程序如下：

```
40 FOR I=1 TO N
45 X=(X-M * Y) * 10
55 M=X/Y:PRINT M;
60 NEXT
```

程序中 45 句的 M 是小数点后上一位的商，N 为要求的精度，N 为多少，就算到多少位。

对于“0”不能做除数的问题，可安排一判断语句进行处理。

```
10 INPUT "X,Y N=";X,Y,N
15 PRINT X"/"Y"=";
20 IF Y=0 THEN PRINT "ERROR! INPUT AGAIN!":GOTO 10
```

判断商的正负，可如下安排语句：

```
25 IF (X>0 AND Y<0)OR(X<0 AND Y>0) THEN PRINT"-";
```

当处理完商的符号后，把被除数、除数均以正数处理。即

```
30 X=ABS(X):Y=ABS(Y)
```

然后利用整数 BASIC 对商去尾取整的特点可方便的求出商的整数部分 M，打印 M，并在 M 后打印 一个小数点。

```
35 M=X/Y:PRINT M;".";
```

达到运算精度后，对最后一位小数进行四舍五入处理。

```
50 IF I=N THEN 60
60 M1=X/Y:IF M1>5 THEN M=M+1
65 PRINT M : END
```

至此，全部程序设计完成，汇总合成后即得完整的除法运算程序。

程序 6—9：简单整数高精度除法运算程序

```
1 REM "No. 6-9 CHU FA"
5 CLS:CLEAR
10 INPUT "X/Y……N=";X,Y,N
15 PRINT X"/"Y"=";
20 IF Y=0 THEN PRINT "ERROR! INPUT AGAIN!":GOTO 10
25 IF (X>0 AND Y<0)OR(X<0 AND Y>0) THEN PRINT "-";
30 X=ABS(X):Y=ABS(Y)
35 M=X/Y:PRINT M;".";:X=(X-M*Y)*10
40 FOR I=1 TO N
45 M=X/Y:X=(X-M*Y)*10
50 IF I=N THEN 60
55 PRINT M;:NEXT
```

```
60 M1=X/Y:IF M1>=5 THEN M=M+1
65 PRINT M:END
```

若要连续运算，读者可在 65 行后加结束处理程序。

程序 6－9 只能计算－32768～32767 之间的整数高精度除法。欲进行任意数高精度除法，仍须借助字符串变量和数组才能实现。由于任意数的高精度除法程序需要处理的问题较多，程序较复杂。因此把该问题单列一节专题讨论。

6.8 任意整数的高精度除法运算

为简化问题，便于讨论，首先介绍任意整数的高精度除法运算程序的编写方法，然后再推广到小数除法运算。

6.8.1 分析问题性质

(1)由于要求任意位整数的高精度除法运算，所以必须采用串变量装数据的方法。

(2)被除数为“0”时，商为“0”；除数为“0”时，打印出错，重新输入数据。

(3)程序中要考虑判别商是正数还是负数的条件语句。

(4)打印输出中，必须考虑商的整数部分和小数部分。

(5)为处理小数点后各位的运算，以达到任意精度的要求，必须考虑不够除补 0 的问题。

(6)为满足任意精度要求，小数点以后商的各位必须正确安排，要求多少位就算到多少位。

(7)由于采用字符串变量装数据，必须考虑由此带来的一些特殊问题。如商的小数点位置的判别及确定。在程序 6－9 中，可由 X/Y 一次确立整数部分，然后点小数点，余数补“0”，计算小数部分，判别小数点比较简单。而使用串变量装数据，则必须判断被除数串与除数串长度的对应情况，仅当被除数串长度短于除数串或

者长度相等，被除数小于除数的情况下，才点小数点等等。这些必须全面考虑。

6.8.2 选取运算方法

使用简单变量装数据比较两数大小较方便，确定商的大小也较容易。如 X＝3245、Y＝579 求 X÷Y＝?，可用 Z＝X/Y 即可求出商 Z＝5。而用串变量装数据，这种方法则行不通。虽然可使用 VAL 函数，把字符变为数字求商，这只能在整数小于 32767 时才可行。现在我们编制的是任意数的除法程序，数据可能是十几位、几十位甚至上百位，以上方法显然不适用。因此，必须寻找可行的运算方法。我们知道在珠算除法中，有一种叫“小扒皮”的运算方法，也叫以减代除运算法。即商 1，在被除数中减一次除数，然后商加 1，再减一次除数，直到不够减时，即得本位之商。这种运算方法浅显易懂，便于掌握，虽然操作繁琐，却正适用计算机运算处理。因此引入以减代除作为解决我们问题的运算方法是最巧妙、最恰当的选择。这样可省去商的数值大小判断程序，只要一次次的从被除数中减去除数，每减一次，记录一次，直到不够减时，减运算的次数就是商，本位商的运算结束。然后被除数增加位数再求第二位商……。采用这种方法可极大地简化运算程序。

6.8.3 确定设计思路

我们首先考察笔算除法的过程：

例：求－122321÷123＝?

```
          9 1 3 . 1
      ┌──────────
123 ) 1 2 2 3 2 1
      1 1 0 7
      ─────────
          1 6 2
          1 2 3
          ─────
            3 9 1
            3 6 9
            ─────
              2 2 0
              1 2 3
              ─────
                9 7
```

把以上过程抽象化，即得以下五个步骤：

设被除数的位数为 A，除数的位数为 B

(1)首先去除两数的符号，再根据两数的符号确定商的符号。

(2)比较被除数 X 左边同除数 Y 相同位数的数(B 个数)122 是否大于 Y(123)，若大于 Y，则在被除数左起第 B 位写商；若小于 Y，则在 X 的左起第 B+1 位写商。

(3)试商。商乘以除数。从被除数的适当位减去商和 Y 的积。所得之差，若小于 Y 则表示试商成功；否则若余数大于 Y 则商再增大；若差小于 0，商再减小。

(4)商的第一位运算结束，把余数与被除数的剩余部分合在一起组成新的被除数(1621)，再进行(2)→(3)→(4)操作。

(5)当最后的被除数小于 Y 时，商的整数部分运算结束。若继续求商，则对商加小数点，被除数补“0”，继续进行(2)→(3)→补“0”的操作，直到商达到要求的精度，才结束运算。(第(5)项操作的前提是余数不为零。)

根据笔算除法的过程，确定了我们的设计思路为：

第一在被除数、除数分别装入串变量 X$、Y$ 后，首先进行数值为 0 的处理，然后再进行去符号处理。

第二设 X$ 的字符长度为 A，Y$ 的字符长度为 B，从 X$ 中取出 B 个字符为 X1$，剩余的为 X2$。

第三从左向右依次比较 X1$、Y$ 中各位数值的大小，若 X1$ 中的大，则进行求商运算。若 X1$ 中的小，则从 X$ 中取 B+1 个字符为 X1$，进行求商运算，剩余的 A−B−1 个字符装入 X2$ 中。

第四本位求商结束后，打印商，把余数装入 Z$ 串变量中，再与 X2$ 相加，组成新的 X$，再进行第二→第三→第四项操作。

第五当 X2$ 的长度为 0 时，则对商添小数点，X$ 加“0”，继

续进行第二→第三→补“0”的操作，直到商达到要求的精度。

以上五点是我们的基本设计思路，当然，实际情况要复杂得多。如被除数小于除数时(A<B)，要首先对商添小数点，被除数补“0”。若 A 仍小于 B，则要对商添“0”，被除数再补“0”，直到 A=B 为止，再进行绝对值的比较。再如，X$ 的前 B 位与 Y$ 相同，以后各位均为 0 的情况，则商 1 后，不再运算，直接对商添“0”等等。这些具体细节，留待编程时讨论。

6.8.4 划分功能模块

根据设计思路，我们可方便地画出功能模块图(见图 6－7)：

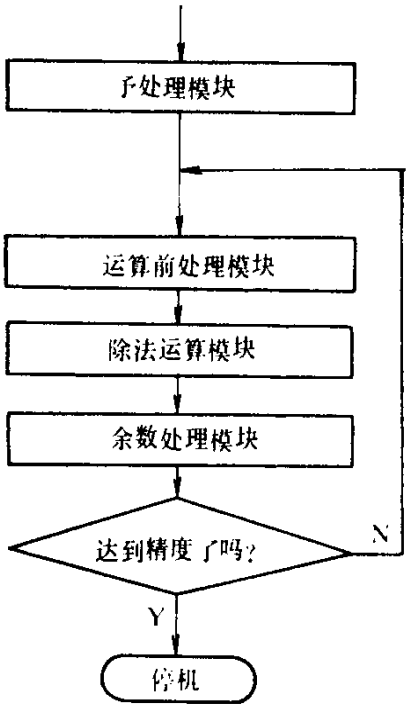


图 6－7　任意整数高精度除法运算程序框图

予处理模块的功能是对被除数、除数为 0 时的处理、去符号、确定商的符号。(若为小数时，去小数点。现在我们编写的是整数程序，可暂不考虑小数。)

运算前处理模块的功能是取被除数 X1$，比较 X1$、Y$ 的数值大小，决定被除数位数的多少及对商添小数点、添 0 处理。

运算模块的功能是完成除法运算及商的输出。

余数处理模块的功能是对余数进行整形、余数为 0 的处理和余数与 X2$ 相加组成新的 X$ 的处理。

6.8.5 绘制程序框图

由于程序比较大，我们按功能模块分别画出程序框图。

(1)予处理程序框图见图 6－8。

(2)运算前处理框图见图 6－9。

(3)除法运算框图见图 6－10。

(4)余数处理框图见图 6－11。

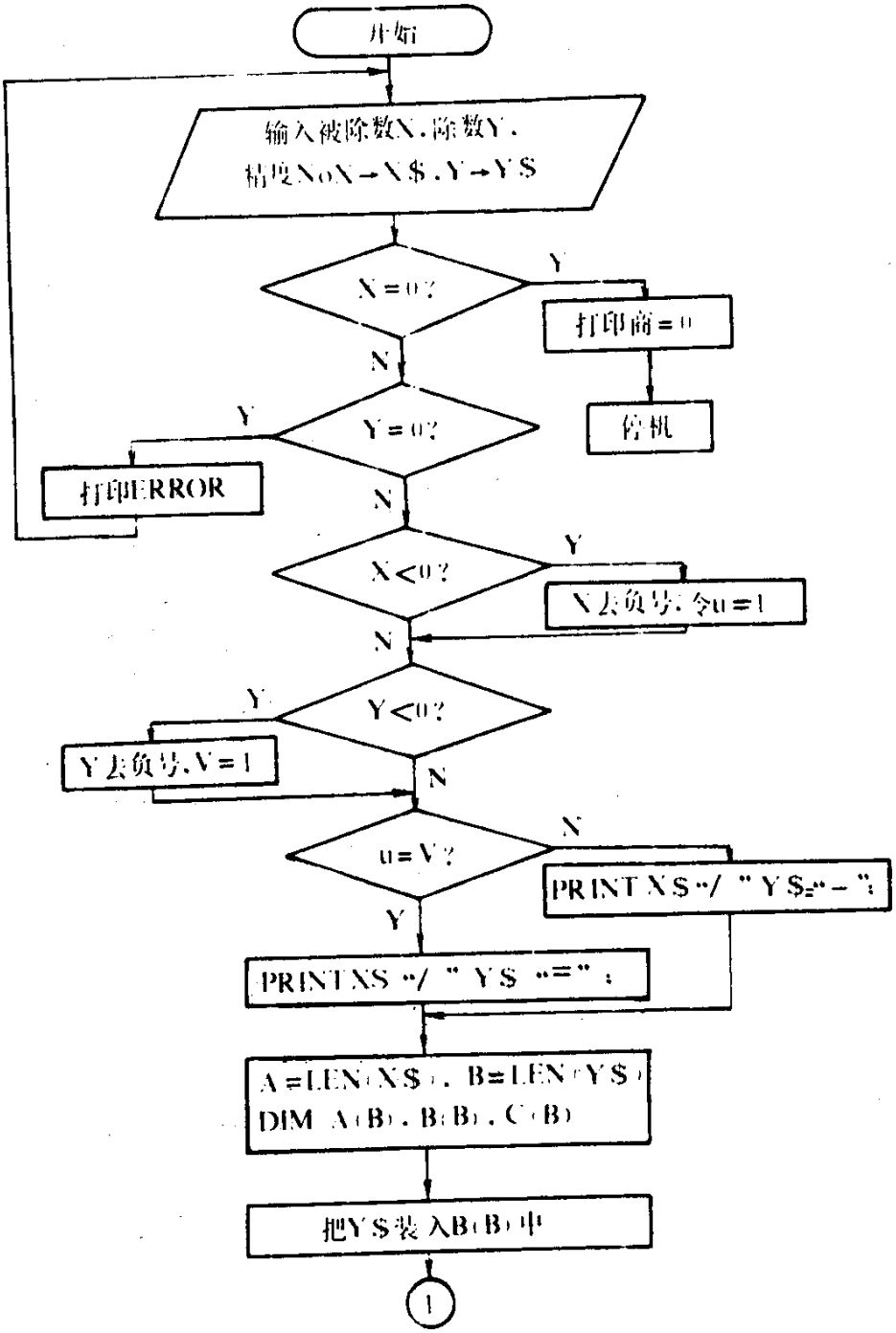


图 6—8　任意整数高精度除法运算程序予处理程序框图

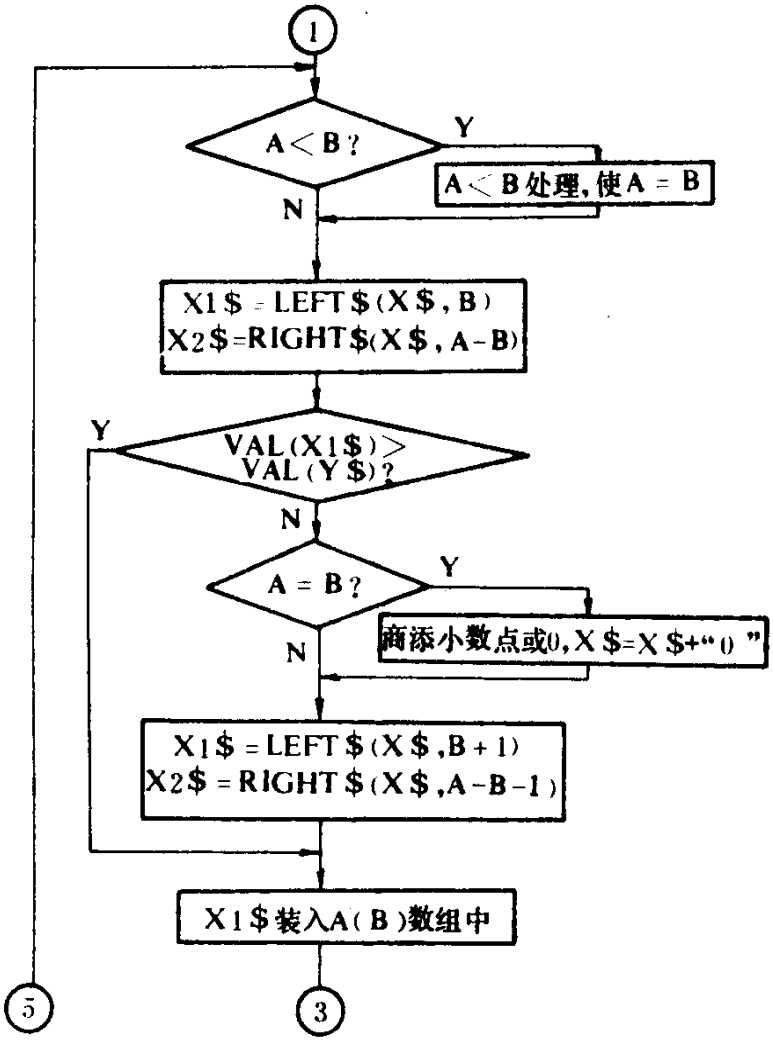


图 6—9　任意整数高精度除法运算程序运算前处理框图

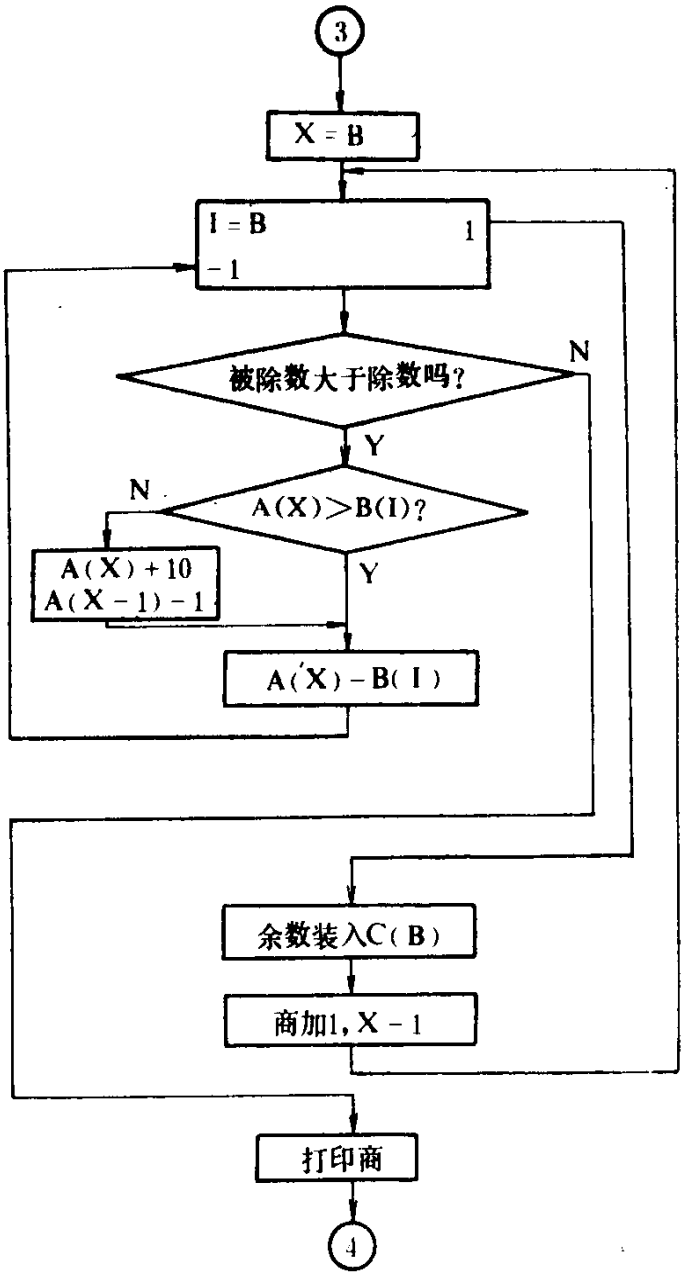


图 6－10　任意整数高精度除法运算程序除法运算框图

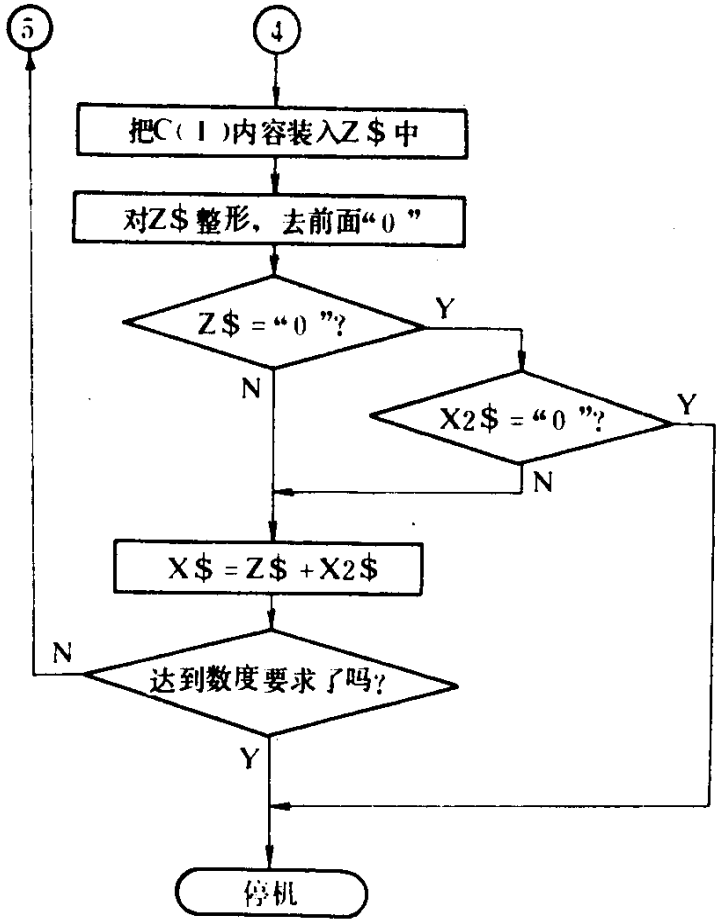


图 6－11　任意整数高精度除法运算程序余数处理框图

6.8.6　分段编写程序

根据程序框图，我们可以编出相应的程序：

1. 予处理程序

这一部分程序比较简单，可以直接编写如下：

```
5 CLS:CLEAR:U=0:V=0
10 INPUT "X/Y……N";X$,Y$,N:PRINT X$"/"Y$"=";:A=
   LEN(X$):B=LEN(Y$)
15 FOR I=1 TO A:XX=VAL(MID$(X$,I,1))
20 IF XX>0 THEN 30
```

```
25 NEXT:PRINT "0":END
30 FOR I=1 TO B:YY=VAL(MID$(Y$,I,1))
35 IF YY>0 THEN 45
40 NEXT:PRINT "ERROR! INPUT AGAIN!": GOTO 10
45 IF LEFT$(X$,1)="-" THEN A=A-1:X$=RIGHT$(X$,
   A):U=1
50 IF LEFT$(Y$,1)="-" THEN B=B-1:Y$=RIGHT$(Y$,
   B):V=1
55 IF U<>V THEN PRINT "-";
60 DIM A(B),B(B),C(B)
65 FOR I=1 TO B:B(I)=VAL(MID$(Y$,I,1)):NEXT
```

程序中 15～25 行判断 X 是否为 0，若 X 为 0，则商等于 0。30～40 行判断 Y 是否为 0，若 Y＝0，则打印出错，重新输入数据。45～60 行判断两数是否为负数，若有一个是负数，则商为负数。65 行把 Y 按高位到低位的顺序依次装入 B(1)～B(B)下标变量中。

2. 运算前处理程序

运算前主要是对被除数和商的处理，这里要分别讨论 A＝B、A＞B、A＜B 三种情况。

(1)当 A＝B 时，先比较|X|、|Y|的大小。若|X|≥|Y|则 X1$＝X$，X2$＝"?"，转求商程序；若|X|＜|Y|，则先打印"0."，对 X 补 0，再转求商程序。

(2)当 A＞B 时，先取 X$ 中的前 B 个字符。即 X1$＝LEFT$(X$,B)，X2$＝RIGHT$(X$,A－B)，转 A＝B 情况的处理。若|X1|＝VAL(X1$)大于|Y|时，则转求商程序。若|X1|＜|Y|，则令 X1$ 多取一位，即 X1$＝LEFT$(X$,B＋1)，X2$＝RIGHT$(X$,A－B－1)，再转求商程序。

(3)当 A＜B 时，则先打印"0."，对 X 补 0，检查是否仍然 A＜B，若是，则商 0，对 X 再补"0"……，直到 A＝B，转 A＝B 情况的处理。这时若|X|＜|Y|，则商 0，对 X 再补 0，转求商程序。

根据以上三种不同情况的分析，可分别写出如下程序：

• A＜B 时的处理程序

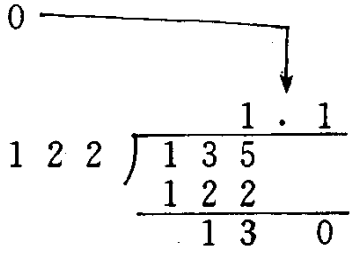
```
125 T=C=D=E=G
130 A=LEN(X$)
135 IF A<B AND G=0 AND C=0 AND D=0 THEN PRINT "0.";:
    X$=X$+"0":A=A+1:C=1:G=1:D=1
140 IF A<B AND C=0 THEN PRINT ".";:X$=X$+"0":A=A+
    1:D=1:C=1
145 IF A<B AND D=0 THEN X$=X$+"0":A=A+1:D=1
150 IF A<B AND D=1 THEN PRINT"0";:X$=X$+"0":C=C+
    1:A=A+1:GOTO 150
```

135～150 行考虑了 A＜B 时的四种不同情况，它们分别对应如下：

(a)135 行的作用：对原始输入的 X，当 A＜B 时，商要在小数点前添“0”，以符合记数习惯。如：12÷122＝0.……，而对再生的(运算后产生的)X，当 A＜B 时，要根据情况，对商添“0.”或只添“.”。因此引入了 G、C、D 三个变量加以限制。G 为原始 X 记录变量，其值为 0，当运算过一次后 G≡1。C 是精度记录变量，它记录小数点后的位数。求商的整数部分时，C＝0，添小数点后 C＞0。D 记录本次运算中 X 是否添过“0”，未添过 D＝0，添过 D＝1。关于 G、C、D 三变量的使用方法和技巧，将在后续程序中体现。

(b)140 行的作用：进行求商运算后，对再生的 X，当 A＜B 时，则对 X 补 0，而商只添小数点。如 135÷122＝1.1……

只添小数点，X 补 0



(c)145 行的作用：在添过小数点，至少进行一次小数商运算后，对再生的 X，当 A＜B 时只对 X 添 0，不对商做任何处理。如

2935÷3225＝0.9……

```
          0 . 9
3 2 2 5 ) 2 9 3 5   0
          2 9 0 2   5
          ---------
            3 2   5 0
                  ↑
                  └只对 X 补“0”
```

(d)150 行的作用：当由 145 行补过 0 仍然 A＜B 时，要对商添“0”，X 再补“0”。若仍然 A＜B 则商再添“0”，X 再补“0”……直到 A＝B。如

2935÷3225＝

```
                  商添“0”
                  ↓
            0 . 9 1 0
3 2 2 5 ) 2 9 3 5   0
          2 9 0 2   5
          ---------
            3 2   5 0
            3 2   2 5
            -------
                  2 5 0
                  ↑
                  └X 补“0”
```

135～150 行进行了 A＜B 的处理，使 A＝B。若 A＞B 则 135～150 行不起作用。因此，运行完 150 行以后一定有 A≥B。这时从 X$ 中取出 B 个字符装入 X1$ 中，进入 A＝B 的操作。程序如下：

- A＝B 时的处理程序

```
155 X1$=LEFT$(X$,B):IF A=B THEN X2$="?":GOTO 165
160 X2$=RIGHT$(X$,A-B)+"?"
165 FOR I=1 TO B:X1=VAL(MID$(X1$,I,1)):Y1=VAL(MID
    $(Y$,I,1))
```

```
170 IF X1>Y1 THEN 185
175 IF X1<Y1 THEN 195
180 NEXT
185 FOR I=1 TO B:A(I)=VAL(MID$(X1$,I,1)):NEXT
190 A(0)=0:GOTO 240
195 IF A>B AND E=1 THEN PRINT"0";:GOTO 225
200 IF A=B AND G=0 AND C=0 AND D=0 THEN PRINT "0.";:
    X$=X$+"0":C=1:D=1:A=A+1
205 IF A=B AND E=1 AND C=0 THEN PRINT "0.";:X$=X$+
    "0":C=1:D=1:A=A+1
210 IF A=B AND C=0 THEN PRINT ".";:X$=X$+"0":A=A+
    1:C=1
215 IF A=B AND D=0 THEN X$=X$+"0":A=A+1
220 IF A=B AND D=1 THEN PRINT "0";:X$=X$+"0":A=A+
    1:C=C+1
```

155～160 行是取被除数 X1$。

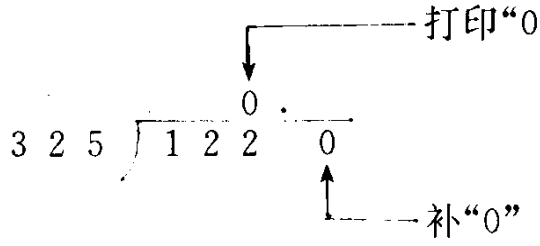
165～180 行是比较被除数和除数绝对值的大小。若被除数大于或等于除数，则把 X1$ 装入 A(B)数组，转运算程序；否则转 A=B、|X|<|Y|的处理程序。

185 行把 X1$ 装入 A(1)～A(B)下标变量中。

195 行判别、进行 A>B 的处理。

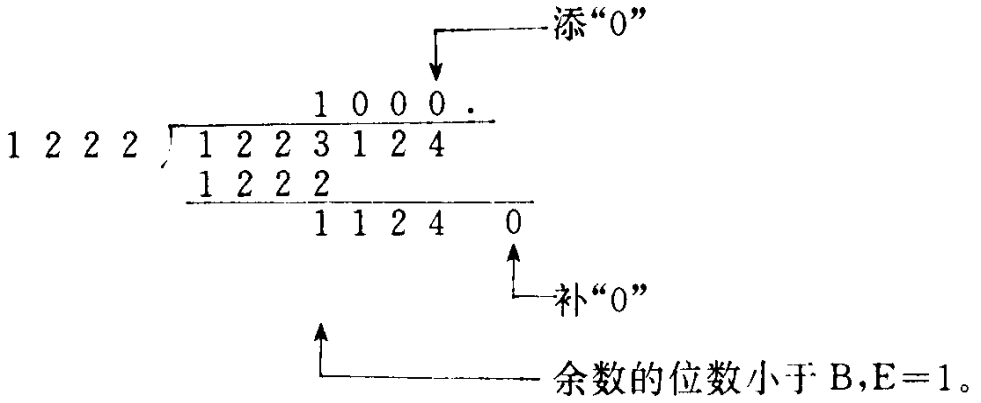
200～220 行考虑了 A=B 时的五种情况的处理。分别解释如下：

(a)200 行的作用：对原始输入的 X，当 A=B、|X|<|Y|时，X 要补“0”，商要打印“0.”，如 122÷325=



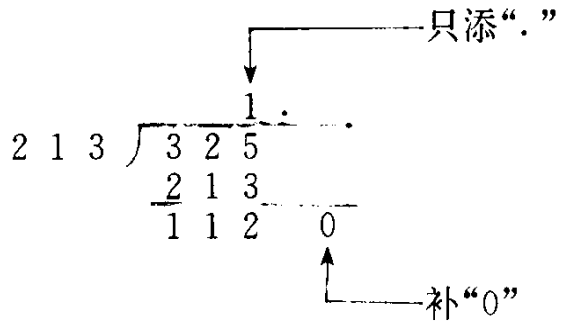
(b)205 行的作用：进行一次除法运算后，对再生的 X，当 A＝B、|X|＜|Y|时商要加“0.”，X 补“0”，这里由 E 和 C 控制该语句是否起作用。C＝0 表示以前进行的运算是求商的整数部分。E＝1 表示余数的位数小于 B。(若等于 B 时，E＝0，E 在余数处理程序中赋值。)

如 1223124÷1222＝



(c)210 行的作用：进行至少一次除法运算后，当余数的位数等于 B，且 X2$＝“?”(即由余数再生 X。)时，只对 X 补“0”，商只添小数点。

如 325÷213＝



(d)215 行的作用：在进行过至少一次商的小数运算后，若余数的位数等于 B(即由余数再生 X)时，只对 X 补“0”，对商不作处

理。

如 321÷221=

```
              1 . 4
 2 2 1 ) 3 2 1
         2 2 1
         1 0 0   0
           8 8   4
           1 1   6 0
                   ↑
                   └── 补"0"
```

(e)220 行的作用：在 A<B 时，通过 145 行或 150 行对 X 补“0”，达到 A=B。若这时的|X|<|Y|，则要对商添“0”，对 X 添“0”。

如 1223÷1222=

```
                       ┌── 添"0"
                       ↓
               1 . 0 0 0
 1 2 2 2 ) 1 2 2 3
           1 2 2 2
               1   0 0 0
                       ↑
                       └── 补"0"
```

通过 200～220 行的处理，达到了 A>B。

- A>B 时的处理程序

A>B 的情况，处理较简单。程序如下：

```
225 A=LEN(X$):X1$=LEFT$(X$,B+1):IF A=B+1 THEN
    X2$="?":GOTO 235
230 X2$=RIGHT$(X$,A-B-1)+"?"
235 FOR I=0 TO B:A(I)=VAL(MID$(X1$,I+1,1)):NEXT
```

235 行完成把 X1$ 中的字符从左向右依次装入 A(0)～A(B)下标变量中。

至此，运算前的处理程序编写完毕。

3. 除法运算程序

根据除法运算程序框图，编制运算程序如下：

```
240 X=B
245 FOR I=B TO 1 STEP-1
250 IF A(0)*10+A(1)<B(1) THEN 275
255 IF A(X)<B(I) THEN A(X)=A(X)+10:A(X-1)=A(X-1)-
    1
260 A(X)=A(X)-B(X):X=X-1:NEXT
265 IF X=0 THEN FOR J=1 TO B:C(J)=A(J):NEXT
270 T=T+1:GOTO 240
275 PRINT T;:E=0:T=0:Z$="0"
```

240～270 行以减代除完成除法运算。

275 行打印输出商。

250 行的作用是判断被除数累减除数后，余数是否还大于除数。若大于除数则继续进行减法操作。由 T 记录减法运算次数，每减一次 T 加 1，一直重复到余数小于除数时，则跳出循环程序，打印输出一位商。

265 行的作用是完整的记录余数。每当一次减法运算完毕，其余数存在 A(1)～A(B)下标变量中，通过 265 行将其转存到 C(1)～C(B)下标变量中。然后 A 数组中的余数再进行下一次减法运算。如此重复直到由 250 行判断余数小于除数时跳出减法运算。这时 C 数组记录的正好是上一次减法运算后的余数。如果不设 265 行而由 A(B)数组输出余数，则有时容易出现错误。例如 4699÷1222：在第一次减法运算后 T＝1，A(B)＝3477，C(B)＝3477，通过 270 行转 240 行；进行第二次减法运算，T＝2，A(B)＝2255，C(B)＝2255，通过 270 行转 240，继续进行第三次减法运算；运算后 T＝3，A(B)＝1033，C(B)＝1033，仍通过 270 行转 240 行进行下一次减法运算；这时 A(1)＝1＝B(1)，则 A(4)＝3－2＝1，A(3)＝3－2＝1，A(2)＝0＜B(2)则通过 255 行 A(2)＝10，A(1)＝0，随

后 A(2)=10－2=8，这时由 250 行判断 A(1)=0<B(1)则跳转 275 行。这时 A(B)=811，若用 A(B)数组输出余数只会输出 811，而不能输出正确的余数 1033。从此可以看出设置 265 行的重要性和必要性。当然也可以不设 265 行而在 270 行中加如下语句

```
270 T=T+1:FOR I=1 TO B:IF A(I)<B(I) THEN 275
273 NEXT
```

同样能达到这一目的。显而易见，要比 265 行复杂一些。

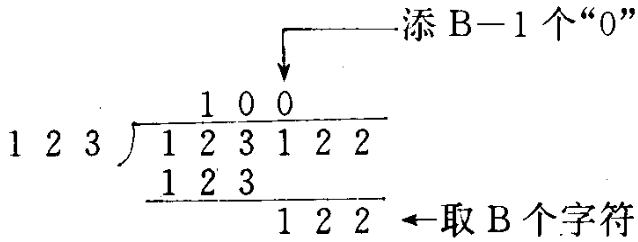
3. 余数处理程序

先分析一下余数的不同形态及其与 X2＄之间的关系。设余数已装入 Z＄中，Z=LEN(Z＄)，X2=LEN(X2＄)

(a)若 Z＄="0"、X2＄="?"，则表示已除尽，应停机。如 123÷123。(对应 285、340 行)

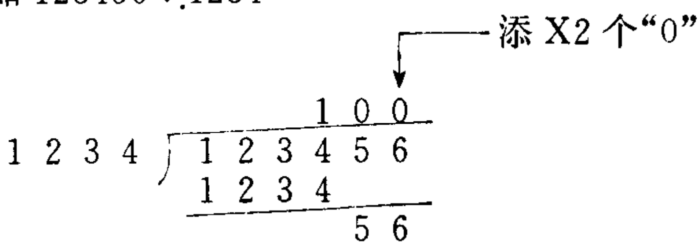
(b)若 Z＄="0"、X2＄="0"，则商要添一个 0，然后停机。如 1230÷123(对应 310 行)。

(c)若 Z＄="0"、X2≥B，则商添 B－1 个 0，Z＄从 X2＄中取 B 个字符。如 123122÷123=



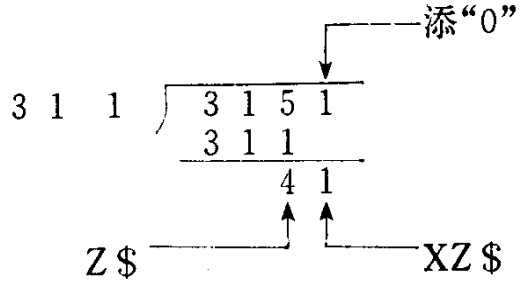
(对应 315、295、300、305 行)

(d)若 Z＄="0"、X2<B，则商添 X2 个 0，Z＄=X2＄、X＄=X2＄。如 123456÷1234=



(对应 320、330 行)

(e)若 X2=1、Z<B-1 则 X$=Z$+X2$,商要添 0。(对应 320 行)如 3151÷311



(f)若 X2=1、Z≥B-1 则 X$=Z$,商不作处理。(对应 325 行)

(g)若 X2>1、Z<B-1 则商添一个 0,Z$ 从 X2$ 中取一个字符,反复进行,直到 Z=B 停止。(对应 320 行)

对以上七种情况的讨论,主要目的是为避免漏掉的整数部分补“0”。该部分程序如下:

```
280 FOR I=1 TO B:Z$=Z$+RIGHT$(STR$(C(I)),1) NEXT
285 IF X2$="?" THEN X$=Z$:GOTO 340
290 Z=LEN(Z$):X2=LEN(X2$):Z=Z-1:Z$=RIGHT$(Z$,
    Z):X2=X2-1:X2$=LEFT$(X2$,X2)
295 IF X2>0 THEN Z$=Z$+LEFT$(X2$,1):Z=Z+1:X2=X2
    -1
300 IF X2>0 THEN X2$=RIGHT$(X2$,X2)
305 IF LEFT$(Z$,1)="0"AND Z>1 THEN Z=Z-1:Z$=
    RIGHT$(Z$,Z):E=1:GOTO 305
310 IF Z=1 AND Z$="0" AND X2=0 THEN PRINT "0";:END
315 IF Z=1 AND Z$="0" AND X2>0 THEN PRINT "0";:GOTO
    295
320 IF X2=0 AND Z>B THEN PRINT "0";:X$=Z$:GOTO 360
325 IF X2=0 AND Z>=B THEN X$=Z$:GOTO 360
```

```
330 IF Z<B THEN PRINT "0";:GOTO 295
335 X$=Z$+X2$
340 A=LEN(X$):IF A=1 AND X$="0" THEN END
345 IF LEFT$(X$,1)="0" THEN X$=RIGHT$(X$,A-1):
    GOTO 340
350 IF C>0 THEN C=C+1
355 IF C>N THEN END
360 D=0:G=1:GOTO 130
```

程序中 280 行完成把余数装入 Z$ 串变量中，285～335 行完成 X 的再生，其中 295～330 行对应以上讨论的七点。350～355 行实现运算精度的控制。

到此为止，四个功能模块的程序编写完毕。

6.8.7 程序的汇总合成

把以上四个功能模块程序合并到一起，就成为一个完整的任意整数高精度除法运算程序。现列出程序清单，见程序 6—10：

程序 6—10：任意整精高精度除法

```
1 REM "NO.6—10 SUP—INT—CHUFA"
2 CLS
5 CLEAR:U=V=T=C=D=E=G=
10 INPUT "X/Y……N=";X$,Y$,N:PRINT X$ "/" Y$"=";:A
   =LEN(X$):B=LEN(Y$)
15 FOR I=1 TO A:XX=VAL(MID$(X$,I,1))
20 IF XX>0 THEN 30
25 NEXT:PRINT "0":END
30 FOR I=1 TO B:YY=VAL(MID$(Y$,I,1))
35 IF YY>0 THEN 45
40 NEXT:PRINT:PRINT "ERROR! INPUT AGAIN!":GOTO 5
45 IF LEFT$(X$,1)="-" THEN A=A-1:U=1:X$=RIGHT
   $(X$,A)
50 IF LEFT$(Y$,1)="-" THEN B=B-1:V=1:Y$=RIGHT
```

```
    $(Y$,B)
55 IF U<>V THEN PRINT "-";
60 DIM A(B),B(B),C(B)
65 FOR I=1 TO B:B(I)=VAL(MID$(Y$,I,1)):NEXT
130 A=LEN(X$)
135 IF A<B AND G=0 AND C=0 AND D=0 THEN PRINT "0.";:
    X$=X$+"0":A=A+1:C=1:G=1:D=1
140 IF A<B AND C=0 THEN PRINT ".";:X$=X$+"0":A=A+
    1:D=1:C=1
145 IF A<B AND D=0 THEN X$=X$+"0":D=1:A=A+1
150 IF A<B AND D=1 THEN PRINT "0";:X$=X$+"0":C=C+
    1:A=A+1:GOTO 150
155 X1$=LEFT$(X$,B):IF A=B THEN X2$="?":GOTO 165
160 X2$=RIGHT$(X$,A-B)+"?"
165 FOR I=1 TO B:X1=VAL(MID$(X1$,I,1)):Y1=VAL(MID
    $(Y$,I,1))
170 IF X1>Y1 THEN 185
175 IF X1<Y1 THEN 195
180 NEXT
185 FOR I=1 TO B:A(I)=VAL(MID$(X1$,I,1)):NEXT
190 A(0)=0:GOTO 240
195 IF A>B AND E=1 THEN PRINT "0";:GOTO 225
200 IF A=B AND G=0 AND C=0 AND D=0 THEN PRINT "0.";:
    X$=X$+"0":C=1:A=A+1
205 IF A=B AND E=1 AND C=0 THEN PRINT "0.";:X$=X$+
    "0":C=1:A=A+1
210 IF A=B AND C=0 THEN PRINT ".";:C=1:X$=X$+"0":A
    =A+1
215 IF A=B AND D=0 THEN X$=X$+"0":A=A+1
220 IF A=B AND D=1 THEN PRINT "0";:X$=X$+"0":A=A+
    1:C=C+1
```

```
225 A=LEN(X$):X1$=LEFT$(X$,B+1):IF A=B+1 THEN
    X2$="?":GOTO 235
230 X2$=RIGHT$(X$,A-B-1)+"?"
235 FOR I=0 TO B:A(I)=VAL(MID$(X1$,I+1,1)):NEXT
240 X=B
245 FOR I=B TO 1 STEP-1
250 IF A(0)*10+A(1)<B(1) THEN 275
255 IF A(X)<B(I) THEN A(X)=A(X)+10:A(X-1)=A(X-1)-
    1
260 A(X)=A(X)-B(I):X=X-1:NEXT
265 IF X=0 THEN FOR J=1 TO B:C(J)=A(J):NEXT
270 T=T+1:GOTO 240
275 PRINT T;:E=0:T=0:Z$="0"
280 FOR I=1 TO B:Z$=Z$+RIGHT$(STR$(C(I)),1):NEXT
285 IF X2$="?" THEN X$=Z$:GOTO 340
290 Z=LEN(Z$):X2=LEN(X2$):Z=Z-1:Z$=RIGHT$(Z$,
    Z):X2=X2-1:X2$=LEFT$(X2$,X2)
295 IF X2>0 THEN Z$=Z$+LEFT$(X2$,1):Z=Z+1:X2=X2
    -1
300 IF X2>0 THEN X2$=RIGHT$(X2$,X2)
305 IF LEFT$(Z$,1)="0" AND Z>1 THEN Z$=RIGHT$(Z$,
    Z-1):Z=Z-1:E=1:GOTO 305
310 IF Z=1 AND Z$="0" AND X2=0 THEN PRINT "0";:END
315 IF Z=1 AND Z$="0" AND X2>0 THEN PRINT "0";:GOTO
    295
320 IF X2=0 AND Z<B THEN PRINT "0";:X$=Z$:GOTO 360
325 IF X2=0 AND Z>=B THEN X$=Z$:GOTO 360
330 IF Z<B THEN PRINT "0";:GOTO 295
335 X$=Z$+X2$
340 A=LEN(X$):IF A=1 AND X$="0" THEN END
345 IF LEFT$(X$,1)="0" THEN X$=RIGHT$(X$,A-1):
```

```
    GOTO 340
350 IF C>0 THEN C=C+1
355 IF C>N THEN END
360 D=0:G=1:GOTO 130
```

6.8.8 讲究设计技巧

通过阅读程序，可以发现程序 6－10 仅使用了几个简单变量，如 A、B、C、D、E、G、U、V、Z 等，通过它们之间不同的逻辑运算较好地处理了除法多变复杂的运算过程。从而有效地解决了高精度除法运算这一难题。又如在运算程序中巧妙地设置了 265 行，引入了余数装载数组 C(B)，较好地解决了余数的保留和转移问题。

6.8.9 检查程序是否通用合理

一个程序编写完成后，要对照设计要求，通过试运行，认真检查、核证，使程序能够适用不同情况，准确无误地反复运行。因此，检查核对过程是必须进行的步骤。

那么应如何检查核对程序呢？通过本例详细介绍检查的方法和过程。

对于本例，首先重点分析讨论除法运算中可能出现的各种情况。其次分析每一种情况是否都能得到正确的处理。最后把每一种情况试运行，检查是否能得出正确的结果。若每一种情况程序都能够正确处理，则程序设计检查通过可以投入使用。

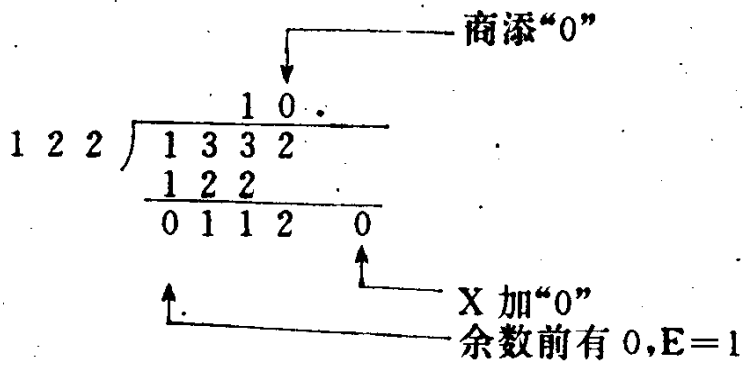
仔细分析除法运算过程，它包含了减、乘运算、试商的逻辑判断、数据的分解合成等多项操作。因此，在算术运算中除法是一个最复杂多变的运算过程。根据以上的分析讨论，归纳起来共有九种形式。现分别检查核对如下：

(a)A＞B、A－B＜B、A 的前 B 位数值大于|Y|、

余数前面有“0”的情况。

对这种情况要对商添“0”，添小数点，X 加“0”。

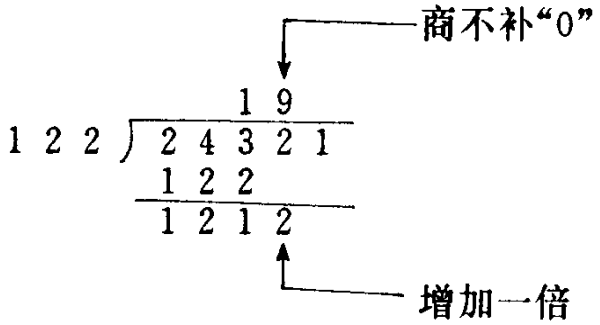
如 1332÷122=



实现过程：通过 295 行，Z $ 从 X2 $ 中取出“2”，这时 Z $ ＝“0112”；通过 305 行记录 E＝1（由于余数前面有“0”）；这时 Z $ ＝“112”、G＝1、D＝0、E＝1、C＝0、A＝B，经 175 行判断 |X1|＜|Y1|，符合 205 行的条件，故商添“0.”，X $ 加“0”。从而实现了商添“0.”，X $ 变为“1120”。

(b)A＞B、A－B＜B、A 的前 B 位数值大于 |Y|、余数前面无 0 的情况。

对这种情况，商不添“0”，仅被除数增加一位，如 24321÷122＝



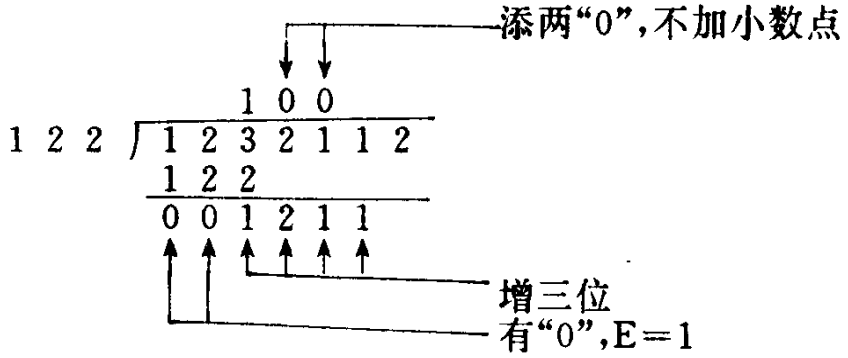
实现过程：通过 295 行、335 行 X $ ＝“12121”；再通过 155 行 X1 $ ＝121；经 175 行判别 |X1|＜|Y1|，转 225 行 X1 $ ＝“1212”；转运算程序，商 9。从而实现了 X $ 增一位，商不补 0。

(c)A＞B、A－B≥B、A 的前 B 位数值大于 |Y|，余数前面有

“0”的情况。

对这种情况商要添“0”；不加小数点；被除数 X1＄增加位数。

如 1232112÷122＝

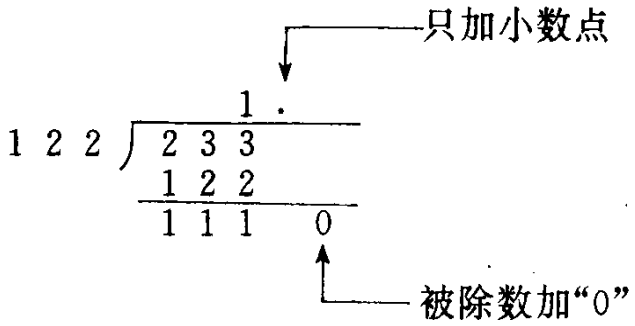


实现过程：通过 305 行 Z＄＝“12”，E＝1；通过 330 行对商添一个“0”；再通过 335 行 X＄＝Z＄＋X2＄＝“12112”。这时 A＞B、E＝1，符合 195 行的条件，故商再添一个“0”；通过 225 行 X1＄＝“1211”转运算程序。从而实现了被除数增加三位，商添两个“0”。

(d)A＝B、|X|＞|Y|、余数前面无“0”的情况。

对这种情况商不添“0”，只加小数点；被除数加“0”。

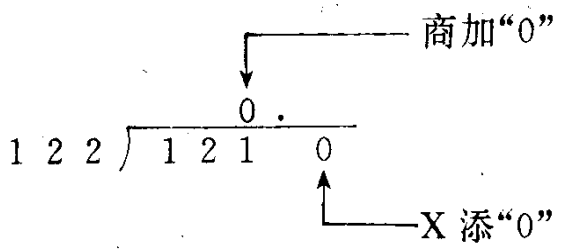
如 233÷122＝



实现过程：通过 285 行 X＄＝Z＄＝“111”，这时 G＝1、A＝B、E＝0、C＝0，符合 210 行的条件，故而打印小数点，X＄加“0”，X＄＝“1110”，转运算程序，从而实现了商只添小数点，X＄加“0”。

(e)A＝B、|X|＜|Y|的情况。

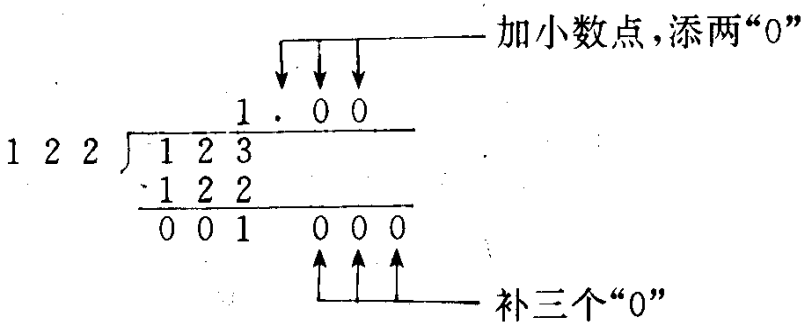
对这种情况，商要加“0”，加小数点。如 121÷122=



实现过程：符合 200 行条件，故商打印“0.”；X$ 加“0”。

(f)A=B、|X|>|Y|、余数前面有两个以上“0”的情况。

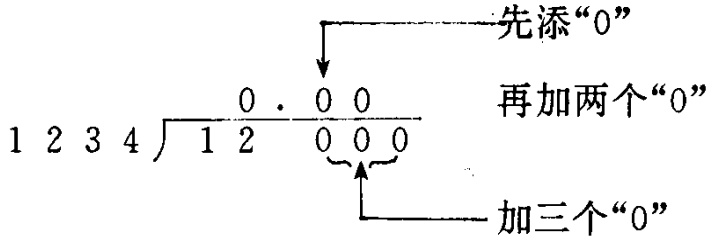
对这种情况商加小数点后，再加“0”，相应被除数也加“0”。如 123÷122=



实现过程：通过 285 行 X$=Z$=“1”，这时 G=1、C=0、D=0、E=0、A<B 符合 140 行的条件，故打印小数点，X$ 加一个“0”，X$=“10”，D=1；C=1，再通过 150 行，对商添“0”，X$ 再添一个“0”，即 X$=“100”；这时符合 220 行条件，故对商再加一个“0”，X$ 再加一个“0”，X$=“1000”；转运算程序。从而实现了商加小数点和两个“0”，对 X$ 补三个“0”。

(g)A<B、A≤B−2 的情况。

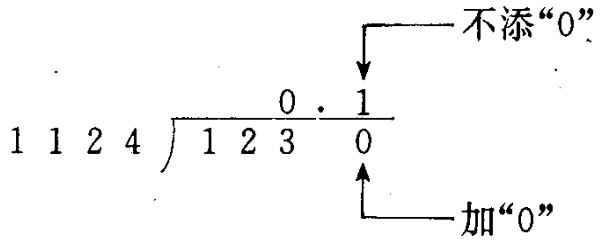
这时商要先添“0.”，再加“0”；X$ 加“0”。如 12÷1234



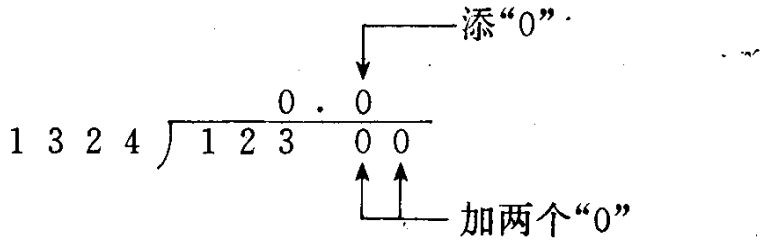
实现过程：通过 135 行先对商打印“0.”，X $ 加一个“0”，X $ ＝“120”，C＝1，G＝1，D＝1；这时符合 150 行条件，对商打印“0”，X $ ＝“1200”；通过 175 行比较，符合 220 行条件，故商再打印一个“0”，X $ ＝“12000”。

(h)A＜B、A≤B－1 的情况。

对这种情况要先对商添“0.”，然后根据 X 的绝对值大于或小于 Y 的绝对值前 A 位的情况，决定对商是添“0”或不添。若|X|大于|Y|的前 A 位则不添“0”，否则添“0”。如 123÷1124＝



再如 123÷1324

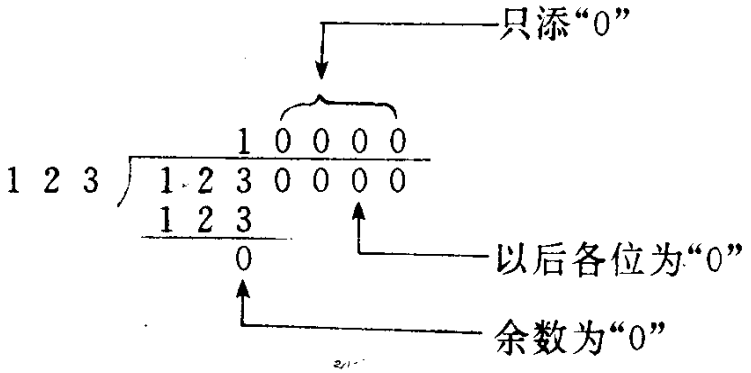


关于实现过程，留给读者检查。

(i)A＞B、A 的前 B 位等于 Y、以后各位均为“0”的情况。

对这种情况在进行一次求商运算后，只对商补“0”，不再运算。

如 1230000÷123=



实现过程：在进行一次除法运算后，通过 295～315 行的三次重复对商打印三个“0”。最后通过 310 行再打印一个“0”后停机。

以上分析检查核对了九种不同情况的运算过程，该程序都能正确运算，得出正确的结果。核查工作可以结束。至此，整个程序编制工作结束。

6.9 任意数的高精度除法

在任意整数高精度除法程序的基础上，再考虑小数的判别和处理，就可编出通用高精度除法运算程序。众所周知，在小数除法运算中有三种形式。第一种被除数是小数，除数是整数；第二种被除数是整数，除数是小数；第三种两数都是小数。实际上还有第四种就是两数都为整数，在程序 10 中已进行了详细的讨论。现在要讨论的是编制任意数（包括小数、整数、负数）高精度通用除法运算程序。

考察笔算小数除法的运算过程，对于小数是如下处理的。对第一种形式我们按整数除法步骤直接进行运算，仅在商的恰当位置添上小数点。对于第二、三种情况，则首先检查除数的小数部分有几位，然后使两数同时扩大相同的倍数，使除数为整数后，再按第一种或整数形式进行除法运算。

参照笔算方法确立编程思路如下：不管那一个数是小数一律化为整数后再套用程序 10 进行除法运算。即对第一种形式对除数添“0”，使被除数去掉小数点；第二种形式对被除数添“0”，除数去小数点；对第三种形式则比较两数小数部分的多少，以小数最多的为参考倍数令两数同时扩大，使两数都去掉小数点。可以看出编程的关键就是把被除数和除数全部化为整数。为此可画出小数化整程序框图，编制小数化整子程序。该子程序计划安排在予处理程序之后，则可以接予处理程序尾编写。图 6－12 画出去小数点程序框图（见图 6－12）。

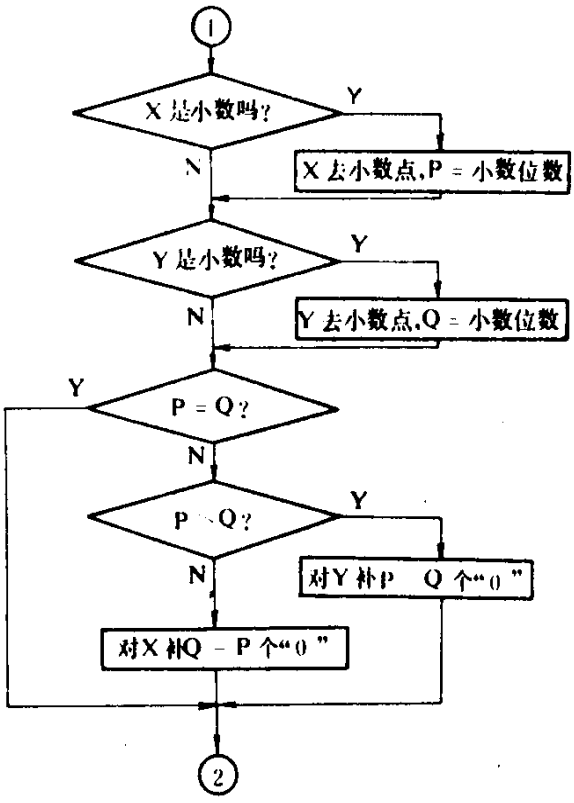


图 6－12　小数除法运算程序去小数点程序框图

根据图 6－12，编制出去小数点子程序如下：

```
60 DELETE 60,65
70 FOR I=1 TO A:IF MID$(X$,I,1)="." THEN P=A-I:GOTO
```

```
80
75 NEXT:GOTO 90
80 A=A-1:X$=LEFT$(X$,A-P)+RIGHT$(X$,P)
85 IF LEFT$(X$,1)="0" THEN A=A-1:X$=RIGHT$(X$,
   A):GOTO 85
90 FOR I=1 TO B:IF MID$(Y$,I,1)="." THEN Q=B-I:GOTO
   100
95 NEXT:GOTO 110
100 B=B-1:Y$=LEFT$(Y$,B-Q)+RIGHT$(Y$,Q)
105 IF LEFT$(Y$,1)="0" THEN B=B-1:Y$=RIGHT$(Y$,
    B):GOTO 105
110 IF P=Q THEN 125
115 IF P>Q THEN FOR I=1 TO P-Q:Y$=Y$+"0":NEXT
120 IF Q>P THEN FOR I=1 TO Q-P:X$=X$+"0":NEXT
125 B=LEN(Y$):DIM A(B),B(B),C(B):FOR I=1 TO B:B(I)=
    VAL(MID$(Y$,I,1)):NEXT
```

程序中 70～80、90～100 两段程序分别去两数的小数点。85、105 两行是考虑当两小数无整数部分去除有效数字左边的“0”。60、125 行是为了与程序 10 相链接。

我们可把图 6－12 与图 6－8～6－11 合在一起，就得到任意数高精度除法运算程序的完整框图。

到此为止，通用高精度除法运算程序全部编写完毕，我们把它排列为程序 6－11：通用高精度除法运算程序。为节约篇幅，不再列程序清单。

6.10 求幂运算

6.10.1 简单整数求幂程序

求幂运算实质上是对一个确定数的连乘运算。我们可以定义一个数组把乘积的各位分别装进数组的各下标变量中，然后每个下标变量中的内容再分别乘以底数，所得之积再存入该数组中，并做好进位处理。如此反复进行，达到指数规定的运算次数后，打印输出结果。这就是本程序的设计思路。读者可自己绘制程序框图。在此仅讨论程序的编写。

设底数为M，指数为N，我们可编一循环程序实现乘积运算。

```
10 DIM A(255):Z=1:C=0:A(1)=1
40 FOR J=1 TO Z
42 A(J)=A(J)*M
45 NEXT
```

运行一次循环后，已将底数M装入A(1)中，A(1)中的数可能大于9，为使每个下标变量中的数小于或等于9，要做好进位的处理。处理程序如下：

```
50 FOR J=1 TO Z
60 IF A(J)>9 THEN B=A(J)/10:A(J)=A(J)-B*10:C=J+1:A
   (C)=A(C)+B:J=C:GOTO 60
70 NEXT
```

进位处理完毕，令Z=C，记录运算次数，再进行第二次求积，再处理进位。求多少次幂就进行多少次循环。把以上两程序合并，使用指数控制循环次数，再加入输入输出语句，就得出完整的求幂程序。

程序6－12：整数求幂运算程序

```
1 REM "NO. 6-12 INT-M∧N"
5 CLS:CLEAR:INPUT "M∧N=";M,N
```

```
10 DIM A(255):Z=1:C=0:A(1)=1
20 FOR I=1 TO N:PRINT "N=";I,
30 IF C>Z THEN Z=C
40 FOR J=1 TO Z:A(J)=A(J)*M:NEXT
50 FOR J=1 TO Z
60 IF A(J)>9 THEN B=A(J)/10:A(J)=A(J)-B*10:C=J+1:A
   (C)=A(C)+B:J=C:GOTO 60
70 NEXT:NEXT
80 PRINT M;"∧"N"=";
90 IF C>Z THEN Z=C
100 FOR I=Z TO 1 STEP-1:PRINT A(I);:NEXT:END
```

说明:20 行根据指数设置循环终值。30、40 行完成乘法运算。50～70 行完成进位处理。80～100 行为结果输出程序。

运行程序。显示

M∧N=?　　输入 12,3↙

N=1　　N=2　　N=3

12∧3=1728

停机。

通过试运行,说明程序编写无误,可以使用。那么其运算能力究竟有多大呢?下面我们看一个实例。

举例:传说古印度国王舍罕在奖赏他的宰相——国际象棋发明者达依尔时,曾问宰相想要什么样的赏赐。达依尔要求赏小麦。他说在象棋的第一格放一粒,第二格放两粒,第三格放四粒,即每一格的粒数是前一格的两倍,只要放满 64 格就心满意足了。国王答应了他的要求。结果全国的小麦用尽也没有填满棋盘。那么,究竟要用多少小麦呢?我们可以运行程序 6—12 算一下:

首先建立解决这一问题的数学模型。设总粒数为 Y,则有

$$Y=2^0+2^1+2^2+2^3+\cdots\cdots+2^{63}=\sum_{n=0}^{63}2^N$$

用数学归纳法可求得

$$\sum_{n=0}^{63} 2^N = 2^{64} - 1$$

运行程序 12，输入 2，64。打印结果为

2∧64＝18，446，744，073，709，551，616

以上结果减 1 即得 Y

Y＝18，446，744，073，709，551，615

若按千粒重 50 克计算，则要用小麦 922 亿吨。难怪印度没有这么多。

再如，假若达依尔要求第二格不是放两粒而是放 1234 粒，以后每格所放的数量是前一格的 1234 倍，那么第 64 格应放多少粒小麦呢？

运行程序 12，输入 1234，63↙打印

1234∧63＝5，660，503，469，520，179，951，666，134，384，010，271，376，949，588，058，518，183，321，697，651，095，960，943，603，301，038，188，497，340，224，785，202，188，794，303，343，668，797，195，134，130，544，516，351，987，929，056，449，089，452，949，737，204，570，484，575，739，878，973，286，261，569，135，57，504

可见$(1234)^{63}$已是一个 195 位的天文数字了。由此可见，使用程序 6－12 已使 F BASIC 语言的幂运算能力得到巨大的增强。但程序 6－12 仍有不足。第一它只能进行 3600 以内任意整数求幂，超过 3600 将产生溢出。原因是程序中使用了变量 M 装底数。3600×9＝32400 已接近整数 BASIC 取值范围的上限。第二不能对小数求幂。如若达依尔要求小麦不是按粒数而是按重量放置，即第一格放 5.678 克，第二格放的重量是第一格的 5.678 倍，余者以此类推。问第 64 格将放多少克小麦。显然，程序 12 是不能运算的。因此必须进一步扩大其求幂能力。实现这一目标仍需使用字符变量。

6.10.2 通用求幂程序

既然求幂运算是对一个确定数的连乘运算，那么我们可以对通用乘法程序进行改造，以实现求幂运算。改造方法如下：

1. 把底数分别装进 B、C 数组，一次求积运算完成后，把积数组(A 数组)的内容分别对应转移给 B 数组，并使 A 数组清零，以便下一次运算装积。

2. 考虑到小数的求幂，必须随时修正小数长度变量。

3. 当运算达到指数要求的次数后，要及时结束运算，输出结果。

改造后的程序如下：

程序 6—13：通用求幂程序

```
5 REM "NO.6-13 SUP-M^N"
10 CLS:CLEAR:DIM A(255),B(255),C(255)
15 INPUT "M^N=";C$,N:PRINT "Please wait....."
20 C=LEN(C$):W=2:B$=C$
25 FOR J=1 TO C:IF MID$(C$,J,1)="." THEN F=J:GOTO 35
30 NEXT:GOTO 40
35 C$=LEFT$(C$,F-1)+RIGHT$(C$,C-F):U=C-F:C=C-1
40 FOR J=1 TO C:IF MID$(C$,J,1)="." THEN NEXT
45 C=C-J+1:C$=RIGHT$(C$,C):G=2*C:B=C:H=2*U
50 FOR J=1 TO C:C(C-J+1)=VAL(MID$(C$,J,1)):B(C-J+1)=C(C-J+1):NEXT
55 FOR K=1 TO C:X=K
60 FOR J=1 TO B:D=B(J)*C(K):A(X)=A(X)+D
65 IF A(X)>9 THEN A(X+1)=A(X+1)+A(X)/10:A(X)=A(X)-A(X)/10*10
70 IF A(A+1)>9 THEN A(X+2)=A(X+2)+1:A(X+1)=A(X+1)-A(X+1)/10*10
75 X=X+1:NEXT
```

```
80 NEXT:W=W+1:IF W>N THEN 105
85 IF A(G)=0 THEN G=G-1
90 FOR I=G TO 1 STEP-1:B(I)=A(I):A(I)=0:NEXT
95 B=G:G=B+C:H=H+U:PRINT "TIME=";W,
100 GOTO 55
105 IF A(G)=0 THEN G=G-1:GOTO 105
110 PRINT:PRINT E$"^"N"=";
115 IF G<H THEN G=H:PRINT "0";
120 FOR I=G TO 1 STEP-1:IF I=H THEN PRINT".";
125 PRINT A(I);:NEXT:PRINT
130 PRINT "Ji Xu Ma? (Y/N)";:Y$=INKEY$(0):IF Y$<>"N"
    THEN 10
135 CLS:LOCATE 10,10:PRINT "GOODBYE!";:END
```

程序说明：

1. 20～50 行对底数去小数点、整形，把底数装入 C、B 数组。其中 W 为循环控制变量。U 为底数小数部分的长度记录变量。H 为幂的小数长度记录变量。每运算一次，W 加 1，H 加 U。

2. 55～80 行为乘法运算程序，与通用乘法程序的对应部分相同。

3. 85～100 行为本程序的关键。它完成了 N－1 次积的转移、乘积数组的清零、幂小数部分长度的计算及控制循环次数，设置循环转移出口。

4. 105～135 行为幂的输出和结束处理程序。

输入程序运行后。显示

M^N=?　　现在输入 5.678,64↙

显示　Please wait……

TIME=3　　TIME=4　　TIME=5……

TIME=64

5. 678^64=1,855,687,507,780,543,495,217,331,136,

461,814,294,200,934,282,131.99745332215575040675574097936734183228405717707553473467130671752118554617810772052147734701982368388061514840288791883164034032674583896466564960091932974973265567162822017722924147918084505 6

$(5.678)^{64}$是一个整数部分有 49 位、小数部分有 192 位、共计 241 位的数字。可见程序 6－13 的幂运算能力已能极好地满足一般运算的需要。这一性能已经远远地超过高档微机。

还有一点，提醒读者注意，程序 6－12、6－13 只能计算结果为 255 位以内的任意数的任意次幂。结果超过 255 位将出现溢出。欲进一步增强运算能力，可把 A、B 数组定义为双下标变量，即 A(X,Y)、B(X,Y)。Y 可选 255，至于 X 的选取理论上可以选 255，可进行结果为 65000 位的幂运算。实际上受微机内存容量的限制选不到这么大。如供用户使用的内存为 8K 字节，则可定义 4000 个下标变量，平均分给 A、B 数则每个数组可定义 2000 个下标变量。因此，应用中要根据内存实际情况合理选取下标。关于双下标变量的使用方法在阶乘运算中介绍。

6.11 阶乘运算

所谓阶乘运算就是把从 1～N 的自然数依次进行乘法运算，一般记作 N!。如求 10 的阶乘则为 10! ＝1×2×3×4×5×6×7×8×9×10。可以看出，求阶乘就是对一组有序的自然数连续求积。因此根据乘法程序的编写方法和求幂运算程序，将很容易编出求阶乘的程序，关键是根据阶数的要求，设置好结束转移出口。由于程序较简单，我们直接编写。见程序 6－14：

程序 6－14：高精度任意阶乘

```
1 REM "NO.6－14 N!"
5 CLS:CLEAR:DIM A(255):A(1)＝1:Z＝1:C＝0:T＝1
```

```
10 INPUT "N! =";N:PRINT "Please wait……"
20 T=T+1:PRINT "TIME="T-1.:IF T>N THEN 100
30 IF C>Z THEN Z=C
40 FOR J=1 TO Z:A(J)=A(J)*T:NEXT
50 FOR I=1 TO Z
60 IF A(I)>9 THEN A(I)=A(I)-A(I)/10*10:C=I+1:A(C)=A
   (C)+B:I=C:GOTO 60
70 NEXT
80 GOTO 20
100 PRINT:PRINT N"! =";
110 IF C>Z THEN Z=C
120 FOR I=Z TO 1 STEP-1:PRINT A(I);:NEXT:END
```

说明：

1. 20 行设置了结束转移出口。当乘完 N 后，转移到结束处理语句。

2. 本程序只能求 130 以内的任意阶乘，这是由于 130 的阶乘已是一个 255 位的数字，已达到 A 数组的下标上限。欲扩大运算能力，可把 A 数组定义为双下标变量。

运行程序，显示

```
N! =?      输入 100↙
Please wait……
TIME=1     TIME=2     TIME=3……
TIME=100
```

100! =93,326,215,443,944,152,681,699,238,856,266,700,490,715,968,264,381,621,468,592,963,895,217,599,993,229,915,608,941,463,976,156,518,286,253,697,920,827,223,758,251,185,210,916,864,000,000,000,000,000,000,000,000

在程序 13、14 的说明中曾提到欲扩大运算能力，需把 A 数组定义为双下标变量。关于双下标变量在程序中应如何安排使用，下

面结合程序 6－15 进行介绍。

程序 6－15：改进型阶乘运算程序

```
1 REM "NO. 6－15 SUP—N!"
5 CLS: CLEAR:C=0:Z=1:T=1:DIM(13,255):A(0,1)=1
10 INPUT: "N! =";N
15 T=T+1:I=0:K=0:PRINT "N="T－1;:IF T>N THEN 70
20 IF C>Z THEN Z=C
25 FOR J=1 TO Z:K=K+1
30 IF K=256 THEN I=I+1:K=1
35 A(I,K)=A(I,K)*T:NEXT
40 I=0:K=0
45 FOR J=1 TO Z
50 K=K+1:IF K=256 THEN I=I+1:K=1
55 IF A(I,K)>9 AND K=255 THEN B=A(I,K)/10:A(I,K)=A(I,
   K)－10*B:C=J+1:A(I+1,1)=A(I+1,1)+B:J=C:GOTO 50
60 IF A(I,K)>9 THEN B=A(I,K)/10:A(I,K)=A(I,K)－10*B:C
   =J+1:A(I,K+1)=B:J=C:GOTO 50
65 NEXT:P=K:Q=I:GOTO 15
70 PRINT N"! =";:K=P:I=Q
75 IF C>Z THEN Z=C
80 FOR J=Z TO 1 STEP－1:PRINT A(I,K);
85 K=K－1:IF K<1 THEN I=I－1;K=255
90 NEXT
95 END
```

说明：

1. A(X,Y)数组下标 X 的选取方法是：先选 X＝0，运行程序，随便计算一个数的阶乘，然后键入 PRINT FRE，检查内存空余单元的数量。例如空余单元为 6200，可以定义 3100 个下标变量，则 M＝3100÷256＝12 余 38，可选取 X＝11。（由于数组下标变量的个数为(X＋1)×(Y＋1)个，所以 X＝M－1＝11。）程序 15 中 A 数

组 X 下标选取 13 是针对科特 FCS－90 学习机的。因为科特供用户使用的内存为 8182，程序占 504 个单元，还剩 7678 个单元，可定义 3839 个下标变量。则 3839÷256＝14 余 155，因此 X 选取 13。

2. 对 A 数组下标变量的使用顺序是 A(0,1)→A(0,255)→A(1,0)→A(1,255)→A(2,0)→……

3. 30、50、85 三行语句是联合使用的。其作用是调整 A 数组第一下标的数值，每当第二下标达到 255 后，则把第一下标加一、第二下标置一。当然也可以把下标变量的使用顺序改为 A(0,0)→A(13,0)→A(0,1)→A(13,1)→A(0,2)→A(13,2)→……。这以各人的喜欢而定。但要注意 30、50、85 三行语句的配套使用。

4. 程序 6－15 未使用 A(X,0)下标变量。该程序可求最大结果为 3570 位的阶乘，故可求 1000 以内的任意阶乘。

6.12 其它数学运算

以上我详细地介绍了加、减、乘、除四种数学运算程序的编写技巧，给出了十五个实用程序的清单。这些程序是我们编制一切运算程序的最基本的工具。其中最重要的是加减混合运算、乘法、除法三个程序(程序 6－5、6－8、6－10)。只要掌握了这些程序，通过灵活地搭配和调用就能编出求根、对数、三角函数等各种复杂的数学运算程序。(使用前要把运算程序转化为具有确定运算功能的子程序)。本文仅以正弦函数运算程序为例，介绍上述运算程序向子程序的转化及主程序对子程序调用、返回等操作的技术处理。给出详细的程序清单。对其它运算程序仅给出粗结构程序，(所谓粗结构是指仅列出算式，不考虑如何计算。)由读者自己完成程序的编写。

6.12.1 正弦函数运算

1. 建立数学模型

根据幂级数展开定理，可把 SIN X 展开为 X 的幂级数，其形式为：

$$SINX=X-\frac{X^2}{3!}+\frac{X^5}{5!}-\cdots\cdots+(-1)^n\frac{X^{2n+1}}{(2n+1)!}+\cdots\cdots$$

可通过求展开式各项的代数和而求出 SIN X 的值。

2. 确定运算方法

一般运算中，对 SIN X 的值精确到小数点后三位即可满足要求。我们选用展开式的前三项即可达到精度要求。则有

$$SINX=X-\frac{X^3}{3!}+\frac{X^5}{5!}\qquad X=\alpha\cdot\frac{\pi}{180}$$

由上式可见，在 SINX 运算中包含了加、减、乘、除、阶乘、求幂多种操作，因而要调用最少四个子程序（加减、阶乘、乘幂、乘法，由于输入角度值要化为弧度值，所以 X 要乘以$\frac{\pi}{180}$）。为减少调用子程序的数量，我们把上式进行改写：

$$SINX=X-\frac{1}{6}X^3+\frac{1}{120}X^5$$ 再改为

$$\begin{cases}X=\alpha\cdot\dfrac{\pi}{180}=\alpha\cdot 0.01745\\ SINX=X-0.16667\cdot X\cdot X\cdot X+0.00833\cdot X\cdot X\cdot X\cdot X\cdot X\end{cases}$$

通过改写只须调用加减和乘法两个子程序就可以完成了。

3. 编写主程序的粗结构程序

```
10 REM "NO. 6－16 SINX"
20 INPUT "X=";X
30 A=X * 0. 01745
40 F=A * A * A
50 G=F * A * A
60 F=F * 0. 16667
70 G=G * 0. 00833
80 H=A－F+G
90 PRINT "SIN X=" H:END
```

粗结构程序给出了SIN X函数值的计算步骤和方法。根据粗结构程序确定要设置加减、乘两个子程序。

4. 编写子程序

我们可把求和通用程序(程序6－5)改为加减子程序。把程序6－8改写为乘法子程序。改写时要去掉输入、演示、延时、打印等语句，并添加运算结果记录、转移语句。各子程序间要注意行号的分配。

(1)加减子程序：输入X$、Y$，输出D$。行号范围200～499。子程序清单如下：

```
200 U$="                    ":N=N1=N2=A=B=C=D=Z=R
    =P=U=V=W:D$="0":X1=LEN(X$):Y1=LEN(Y$)
210 IF LEFT$(X$,1)="-" THEN U=1:X1=X1-1:X$=
    RIGHT$(X$,X1)
215 IF LEFT$(Y$,1)="-" THEN V=1:Y1=Y1-1:Y$=
    RIGHT$(Y$,Y1)
220 FOR I=1 TO X1-1:IF MID$(X$,I,1)="." THEN N1=I:
    GOTO 235
230 NEXT:X$=X$+".0"
235 FOR I=1 TO Y1-1:IF MID$(Y$,I,1)="." THEN N2=I:
    GOTO 245
240 NEXT:Y$=Y$+".0"
245 IF N2<N1 THEN 260
250 IF N2=N1 THEN P=N1:GOTO 265
255 X$=LEFT$(U$,N2-N1)+X$:P=N2:GOTO 265
260 Y$=LEFT$(U$,N1-N2)+Y$:P=N1
265 N1=LEN(X$)-P:N2=LEN(Y$)-P
270 IF N2<N1 THEN 285
275 IF N1=N2 THEN R=N1:GOTO 290
280 X$=X$+LEFT$(U$,N2-N1):R=N2:GOTO 290
285 Y$=Y$+LEFT$(U$,N1-N2):R=N1
```

```
290 N=P-R
292 FOR I=1 TO N:AA=VAL(MID$(X$,I,1)):BB=VAL(MID
    $(Y$,I,1)):IF AA>BB THEN 300
294 IF AA<BB THEN SWAP X$,Y$:W=1:GOTO 300
296 NEXT:IF U<>V THEN D$="0":RETURN
300 IF U=V THEN GOSUB 360
305 IF U<>V THEN GOSUB 400
310 IF R$(N)="." THEN N=N-1:GOTO 320
315 IF R$(N)="0" THEN N=N-1:GOTO 310
320 K=0:D$="0"
325 K=K+1:IF R$(K)="0" THEN 325
330 IF R$(K)="." THEN R$(K)="0."
332 IF (U=1 AND V=1)OR(U=0 AND V=1 AND W=1)OR(U=
    1 AND V=0 AND W=0) THEN D$="-"
335 FOR I=K TO N:D$=D$+R$(I):R$(I)+"":NEXT
340 D=LEN(D$):IF LEFT$(D$,1)="0" THEN D$=RIGHT
    $(D$,D-1)
345 D$=LEFT$(D$,9):RETURN
360 FOR I=N TO 1 STEP-1
365 A=VAL(MID$(X$,I,1)):B=VAL(MID$(Y$,I,1)):Z=A
    +B+C:C=0
370 IF Z>9 THEN C=1
375 R$(I)=RIGHT$(STR$(Z),1)
380 IF I=P+1 THEN I=I-1:R$(I)="."
385 NEXT
390 IF C=1 THEN R$(0)="1"
395 RETURN
400 FOR I=N TO 1 STEP -1
405 A=VAL(MID$(X$,I,1)):B=VAL(MID$(Y$,I,1)):IF A<
    (B+C) THEN Z=A+10-B-C:C=1:GOTO 415
410 Z=A-B-C:C=0
```

```
415 R$(I)=RIGHT$(STR$(Z),1)
420 IF I=P+1 THEN I=I-1:R$(I)="."
425 NEXT:RETURN
```

(2)乘法子程序。输入B$、C$，输出D$，行号范围500～699。程序清单如下：

```
500 E=F=H=D=K=G=U=V=D$="0":B=LEN(B$):C=
    LEN(C$)
510 IF LEFT$(B$,1)="-" THEN U=1:B=B-1:B$=RIGHT
    $(B$,B)
515 IF LEFT$(C$,1)="-" THEN V=1:C=C-1:C$=RIGHT
    $(C$,C)
520 IF U<>V THEN D$="-"
525 FOR I=1 TO B:IF MID$(B$,I,1)="." THEN E=I:GOTO
    535
530 NEXT:GOTO 540
535 B$=LEFT$(B$,E-1)+RIGHT$(B$,B-E):H=B-E:B=
    B-1
540 FOR J=1 TO C:IF MID$(C$,J,1)="." THEN F=J:GOTO
    550
545 NEXT:GOTO 555
550 C$=LEFT$(C$,F-1)+RIGHT$(C$,C-F):H=H+C-
    F:C=C-1
560 FOR I=1 TO B:IF MID$(B$,I,1)="0" THEN NEXT
565 B=B-I+1:B$=RIGHT$(B$,B)
570 FOR J=1 TO C:IF MID$(C$,J,1)="0" THEN NEXT
575 C=C-J+1:C$=RIGHT$(C$,C)
580 G=B+C
590 FOR I=B TO 1 STEP -1:B(I)=VAL(MID$(B$,B-I+1,
    1)):NEXT
595 FOR J=C TO 1 STEP -1:C(J)=VAL(MID$(C$,C-J+1,
```

```
    1)):NEXT
600 FOR K=1 TO C:X=K
605 FOR J=1 TO B:A(X)=A(X)+B(J)*C(K)
610 IF A(X)>9 THEN A(X+1)=A(X+1)+A(X)/10:A(X)=A
    (X)-A(X)/10*10
615 IF A(X+1)>9 THEN A(X+2)=A(X+2)+1:A(X+1)=A(X
    +1)-A(X+1)/10*10
620 X=X+1:NEXT
625 NEXT
630 IF A(G)=0 THEN G=G-1
635 IF G<H THEN G=H:D$=D$+"0"
645 FOR I=G TO 1 STEP -1:IF I=H THEN D$=D$+"."
650 D$=D$+RIGHT$(STR($(A(I)),1):A(I)):A(I)=0:B(I)
    =0:C(I)=0:NEXT
655 D=LEN(D$):IF LEFT$(D$,1)="0" THEN D=D-1:D$=
    RIGHT$(D$,D)
660 IF LEFT$(D$,1)="." THEN D$="0"+D$
665 D$=LEFT$(D$,9):RETURN
```

5. 在主程序中安排对子程序的调用

主程序中 30 行～70 行要调用乘法子程序，80 行要调用加减子程序。调用的方法是：

(1)调用前先对子程序的运算数据赋值。如加减子程序中要对 X$、Y$ 赋值；乘法子程序中要对 B$、C$ 赋值。

(2)调用子程序后，要把子程序的运算结果保存在主程序之中，供主程序使用。具体调用程序安排如下：

```
30 B$=STR$(X):C$="0.01745":GOSUB 500:A$=D$;
40 B$=A$:C$=A$:GOSUB 500:B$=D$:C$+A$:GO-
   SUB500:F$=D$
50 B$=F$:C$=A$:GOSUB 500:B$=D$:C$=A$ GOSUB
   500:G$=D$
```

```
60 B$=F$:C$="0.16667":GOSUB 500:F$=D$
70 B$=G$:C$="0.00833":GOSUB 500: G$=D$
80 X$=A$:Y$="-"+F$:GOSUB 200:X$=D$:Y$=G$:
   GOSUB 200:H$=D$
```

可见，只要设计好子程序，主程序的编写调用都很简单。不过在主程序中要注意处理好以下几个问题：

①从上述程序可以看出，30 行运算一次后，A$ 中装的数据是一个小于 1 的五位小数，若经 40、50 行对 A$ 内的数求五次幂，则其结果将变为小于 1 的 30 位的小数。由于我们只求精确到三位小数的 SINX 的值，所以，保留 30 位小数已没有价值，反而减慢了运算速度。一般对参与运算的数保留五位小数即可达到精度要求。因此每运算一次后要把小数点第七位以后的数删去。可编一子程序进行删数操作。权且称为精度控制子程序。清单如下：

```
100 D=LEN(D$)
110 FOR I=D TO 1 STEP-1:IF MID$(D$,I,1)="." THEN T=
    I:GOTO 130
120 NEXT
130 D$=LEFT$(D$,T+7):RETURN
```

上述程序为通用精度控制程序。对于正弦函数，其值总是小于或等于 1，不用考虑整数部分的判别，所以可直接把 D$ 写为 D$=LEFT$(D$,9)，即从 D$ 中取左边的九位，(其中包括小数点、整数部分的 0 或 1、七位小数)。因此我们对加法子程序中的第 345 行、乘法子程序中的 665 行分别添了 D$=LEFT$(D$,9)。

②对于正弦函数有 SIN(0°)=0、SIN(90°)=1 因此要对这两种极限情况进行予处理。

③当角度小于 7°时，其正弦值与弧度值差别不大。故对于小于 7°的情况，可直接用弧度值代替正弦值以提高运算速度。

①在正弦函数运算中有

SIN(90～180°)=SIN(90～0°)

SIN(180～270°)＝－SIN(0～90°)

SIN (270～360°)＝－SIN(90～0°)

只要知道了 0～90°的正弦值就可知道其它角度的正弦值。因此在主程序中应安排相应的判断处理语句。

根据以上讨论，编写主程序如下：

```
1 REM "NO. 6－17 Sin(x)"
5 CLS:CLEAR:DIM A(50),B(50),C(50):DIM R$(50):Q=0
10 PRINT "IN.X,OUT Sin(X),X>=0,X<32767":INPUT "X=";
   Y:YY=Y
15 IF Y<0 THEN PRINT "INPUT AGAIN!!!":GOTO 10
20 IF Y>360 THEN Y=Y MOD 360
25 IF Y>90 AND Y<=180 THEN Y=180－Y
30 IF Y>180 AND Y<=270 THEN Y=Y－180:Q=1
35 IF Y>270 AND Y<=360 THEN Y=360－Y:Q=1
40 IF Y=0 THEN H$="0":GOTO 100
45 IF Y>87 AND Y<=90 THEN H$="1":GOTO 100
50 IF Y<7 THEN B$=STR$(Y):C$="0. 01745":GOSUB 500:H
   $=D$:GOTO 85
55 B$=STR$(Y):C$="0. 01745":GOSUB 500:A$=D$:
   PRINT:PRINT "Please wait……1";
60 B$=A$:C$=A$:GOSUB 500:B$=D$:C$=A$:GOSUB
   500:F$=D$:PRINT "2";
65 B$=F$:C$=A$:GOSUB 500:B$=D$:C$=A$:GOSUB
   500:G$=D$:PRINT "3";
70 B$=F$:C$="0. 16667":GOSUB 500:F$=D$:PRINT "4";
75 B$=G$:C$="0. 00833":GOSUB 500:G$=D$:PRINT "5";
80 X$=A$:Y$="－"+F$:GOSUB 200:X$=D$:Y$=G$:
   GOSUB 200:H$=D$:PRINT:PRINT
85 H$=LEFT$(H$,5)
90 IF Q=1 THEN H$="－"+H$
```

```
100 PRINT "Sin(" YY")="H$:END
```

6. 程序合成

在子程序的编写中，已经合理地分配了行号，读者只要把主程序和两个子程序合在一起就得出完整的正弦函数运算程序。合成时要注意两点：第一，由于数组不能反复定义，因此必须把各子程序中的数组定义语句移到主程序中（在子程序的编写中已剔除了DIM 语句，集中安置在主程序的第五行。）。第二，，调用前要对各子程序变量清零。由于这时已不能使用CLEAR 语句，所以均采用了对变量置 0 的方法。见 200、500 行。第三，要兼顾主程序、子程序变量的使用。尽量避免使用重复变量，以免出现运算错误。如 600 行中有变量 X，所以在主程序中 SINX 改为 SINY。请读者自己合成程序。不再赘述。

说明：程序 6－17 不能计算负角度的正弦值。若需要，读者可自己添加。实质上，负角度与正角度的正弦值绝对值相等只是符号不同。这只要对正角度函数值前加负号即可解决。

6.12.2 余弦函数运算

在三角函数的运算中，只要求一种函数的值，就可推出其它所有函数的值。如已知 sinX 的值，可通过 $\sin^2X+\cos^2X=1$ 求出 cosX；进而推出 tgX 等。也可以编一程序，直接计算余弦函数的值。方法如下：

余弦函数的幂级数展开式为

$$COSX=1-\frac{X^2}{2!}+\frac{X^4}{4!}-\cdots\cdots+(-1)^n\frac{X^{2n}}{(2n)!}+\cdots\cdots$$

取前三项的和为 COSX 的近似值。则有

$$COSX=1-\frac{X^2}{2!}+\frac{X^4}{4!}$$

$$COSX=1-0.5X\cdot X+0.04167X\cdot X\cdot X\cdot X$$

因此可模仿 SINX 编出 COSX 的粗结构程序。

```
10 REM "NO. 6－18 COS(X)"
20 INPUT "X=";X
```

```
30 A=X*0.01745
40 F=A*A:G=F*F
50 F=0.5*F
60 G=G*0.04167
70 H=1-F+G
80 PRINT "COS("X")=" H:END
```

6.12.3 求根、求幂程序

程序 6－13 是对任意数求任意整数次幂，不能进行指数是小数或指数小于 1 的幂运算。实质上求根、求幂是同一种运算。当指数大于 1 时称为求幂（如 X^2）；指数大于 0 小于 1 时称为求根（如 $X^{0.5}=\sqrt{X}$）；指数小于 0 时则是对幂取倒数（如 $(X)^{-N}=\frac{1}{X^N}$）。因此编一通用程序就可解决求根、求幂两种运算。

根据牛顿二项式定理，对 $(1+X)^A$ 有如下展开式：

$$(1+X)^A=1+A\cdot X+\frac{A(A-1)}{2!}X^2+\cdots\cdots+\frac{A(A-1)\cdots\cdots(A-n+1)}{n!}X^n$$

据此可编制如下粗结构程序：

```
5 REM "NO.6-19 SUP-QIU MI"
10 CLS:CLEAR:INPUT "X∧A";X1,A
20 X=X1-1:Y=1+A*X:N=1:B=A
30 FOR I=1 TO 100
40 N=N*(I+1):B=B*(A-I):X=X*X
50 Z=B*X/N:Y=Y+Z:NEXT
60 P
RINT X1"∧"A"="Y:END
```

说明：(1)该程序要调用除法子程序

(2)30 行中的循环终值越大结果越精确。程序 6－19 堪称得上是万能求幂程序。

6.12.4 求自然对数

对于自然对数 Ln(1+X)有如下展开式：

$$Ln(1+x)=X-\frac{X^2}{2}+\frac{X^3}{3}-\frac{X^4}{4}+\cdots\cdots(-1)^{n-1}\frac{X^n}{n}+\cdots\cdots$$

这个级数收敛较慢，精确到小数四位要求10000项。因此引入了

$$Ln(\frac{N+1}{N})=2(\frac{1}{2N+1}+\frac{1}{3}\cdot\frac{1}{(2N+1)^3}+\cdots\cdots+\frac{1}{2n+1}\cdot\frac{1}{(2N+1)^{2n+1}}+\cdots\cdots)$$

$n=0,1,2,\cdots\cdots$

令 $X=\frac{N+1}{N}$，则有 $N=\frac{1}{X-1}$，$2N+1=\frac{X+1}{X-1}$，$\frac{1}{2N+1}=\frac{X-1}{X+1}$

$$Ln(X)=2[\frac{X-1}{X+1}+\frac{1}{3}\cdot(\frac{X-1}{X+1})^3+\cdots\cdots+\frac{1}{2n+1}\cdot(\frac{X-1}{X+1})^{2n+1}+\cdots\cdots]$$

根据上述展开式可编制如下程序：

```
5 REM "NO. 6-20 Ln(X)"
10 CLS:CLEAR:INPUT "Ln(X),X=";X
20 A=(X-1)/(X+1):B=A:C=A
30 FOR I=3 TO 1000 STEP 2
40 B=B*A*A
50 C=C+1/I*B
60 NEXT
70 C=2*C
80 PRINT "Ln("X")="C:END
```

6.12.5 求 e^x

e^x 函数在应用数学中经常使用，下面给出实现的方法。

e^x 函数的幂级数展开式为

$$e^x=1+X+\frac{X^2}{2!}+\cdots\cdots+\frac{X^n}{n!}+\cdots\cdots(-\infty<X<\infty)$$

对应的粗结构程序为

```
5 REM "NO. 6-21 e∧x"
10 CLS:CLEAR:INPUT "e∧x,x=";X
20 C=1:A=1
30 FOR I=1 TO 100
40 A=A*X:B=1/I!
```

```
50 C=C+A*B
60 NEXT
70 PRINT "e∧"X"=" C:END
```

到此为止，我们讨论了五种较复杂运算程序的编写思路，读者可举一反三编出其它数学运算程序，本节不再一一列举。

第七章　游戏程序的编写技巧

许多玩过电视游戏的朋友提出这样的疑问："丰富多彩的游戏画面是怎样形成的？游戏里的主人翁为什么会受操纵器的控制？一盒录像带仅能放像 2～3 个小时，而那么小的游戏卡为什么放起来没有完？……"学习了上篇内容后，基本明白了"游戏"是通过编制程序实现的。不由得跃跃欲试，想自己动手编一个游戏程序。但是，"想时容易做时难"，多次尝试，都不成功，实在令人沮丧。实际上，游戏程序有它独特的程序结构和编写技巧，有着普通程序中用不到的特殊技术处理。不得要领，贸然编程，即使有经验的编程老手，也往往走弯路，更何况我们初学者呢。

欲编好游戏程序，首先要构思故事情节、确定游戏的内容；其次策划游戏结构、程序结构、设计程序框图；最后编写详细程序、上机操作、运行调试，还要进行程序的修改和维护等工作。总之，编写游戏程序并不难，只要掌握了编程套路，定能得心应手地编出称心如意的游戏程序。

本章拟分"游戏程序的概念、故事情节和游戏结构、程序结构和程序框图、游戏程序的设计过程、游戏程序的编程特技、游戏程序的实例分析"六节，深入浅出，由简单到复杂，通过"成龙救金凤"游戏程序的编写过程，介绍使用 F BASIC 语言编写游戏程序的方法和技巧。同时，每一节还将穿插介绍部分简单游戏程序供大家练习。从而使读者逐步弄通游戏程序的结构，逐步掌握游戏程序的设计过程及功能模块的划分、程序的编写、子程序的调用和链接技术，最终亲手编出满意的游戏程序。最后再介绍几个游戏实例，以

供练习。

7.1 游戏程序的概念

下面我们先通过两个简单的游戏实例认识游戏程序。

7.1.1 《猜数游戏》介绍

例 7.1 《猜数游戏》。这是一个智能类游戏。

本游戏由电脑产生一个随机数字，告诉你猜数的数值范围和最多允许试猜的次数，由游戏者使用键盘输入方式猜数。若在规定的次数猜中，则奏一段明亮的音乐向你祝贺，然后进入下一关。若猜不中，则播放一段低沉的音乐，并提示所猜的数值与谜底数值的比较情况，告诉你猜小一点或大一点的数。若在规定的次数没有猜中，则询问是否继续游戏。你共有三次机会，(即通常游戏中的接关)若三次都没有猜中，则游戏结束。本游戏共 11 关，猜数的范围一关比一关大，难度越来越高。你若能连破 11 关，则高奏欢快的乐曲，祝贺你的成功。

1. 运行程序，在一段欢快的音乐声中显示出标题画面(见图 7—1)。

```
* * * * * * * * * * * * * * * * * *
*          GUESS NUMBER           *
* * * * * * * * * * * * * * * * * *
          Please select:
          STAGE 1--11? □
```

图 7—1

画面为蓝色背景，黄色标题。标题为“猜数游戏”，第二行提示“请选择”，第三行表示从 1～11 关中任选一关。若选第 3 关，则按数字健“3”，回车后则清屏、显示游戏画面(图 7—2)：

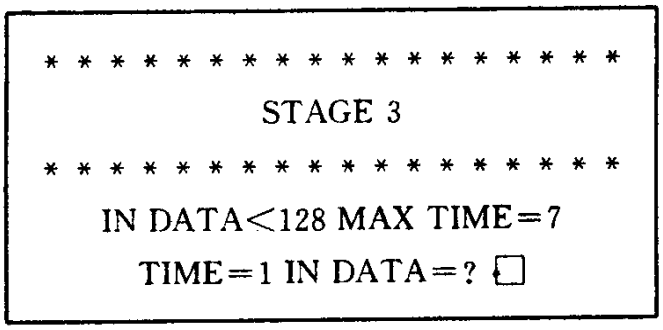


图 7—2

图 7—2 表示：现在是第三关。猜数范围小于 128。最多允许猜 7 次。现在是第一次。请输入所猜的数据。

假定电脑选中的数是 70，你猜的是 80，则演奏低沉的音乐，在屏幕底行显示：

80 is big! Give a smaller data.

提示“80 大了，给一个小一点的数。”这时，输入行变为

TIME=2 IN DATA=? □

如果你在第二次猜 60，回车后，仍奏低沉的音乐，在屏幕底行显示：

60 is small! Give a bigger data.

提示“60 小了，给一个大一点的数。”这时，输入行变为：

TIME=3 IN DATA=? □

“现在是第三次试猜，请输入猜数”。

……

如果你猜中了 70，回车后，则演奏欢快的音乐，清屏，显示红色的

YOU ARE RIGHT !!!

PLEASE GO ON TO STAGE 4

意思是：“你猜对了，请进入第 4 关”。暂停一段时间，清屏。显示图 7—3 的画面，

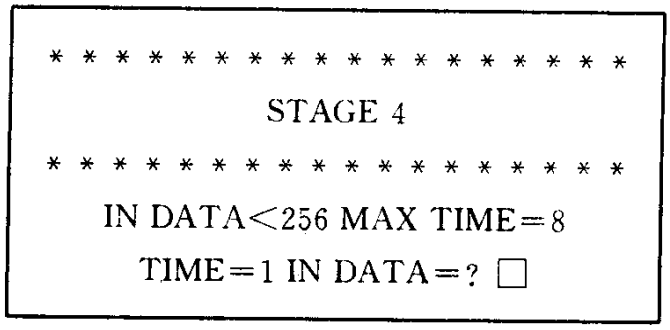


图 7—3

现在进入第 4 关，与第三关相比，猜数范围增大了一倍，允许猜的次数增加了一次。只要在允许试猜的次数之内猜中，都可顺利的进入下一关。

2. 如果你在规定的试猜次数内没有猜中，则清屏，显示：

TIME IS OVER 8 !!!

CONT or END? (Y/N)□

表示“试猜次数超过 8 次，继续或者结束游戏。”若继续游戏按“Y”键；若结束游戏则按“N”键。若欲接关游戏，按“Y”键后，则清屏，显示图 7—3 的画面。可接第四关继续游戏。若欲结束游戏则按“N”键，这时清屏，显示：

GOOD—BYE !!!

结束程序运行。

3. 如果你连破 11 关，取得了最后的胜利则清屏。打印金黄色的

CONGRTULATIONS!!!

意思是“祝贺你取得成功!!!”高奏一曲欢快的乐曲后显示

GOOD—BYE!!!

结束程序运行。

7.1.2 《猜数游戏》程序清单

《猜数游戏》程序清单如下：

```
5 REM "No. 7—1 GUESS NUMBER"
10 CLS:CLEAR:CGEN(2):CGSET 0,1:A=32:B=5:G=0:A$=
```

```
"* * * * * * * * * * * * * * * * * * * *"
15 FOR I=8 TO 20:COLOR I,6,2:NEXT
20 LOCATE 5,4:PRINT A$:LOCATE 5,6:PRINT "* GUESS
   NUMBER *":LOCATE 5,8:PRINT A$:LOCATE 7,12:PRINT
   "Please select:"
25 LOCATE 7,16:PRINT "STAGE 1-11 ";:GOSUB 250
30 INPUT E:IF E=11 THEN A=32767:B=15:GOTO 70
40 IF E=1 THEN 70
50 FOR I=2 TO 10:A=A+A:B=B+1:IF I=E THEN 70
60 NEXT
70 X=RND(A):F=0:IF X=0 THEN 70
80 CLS:LOCATE 5,4:PRINT A$:LOCATE 9,6:PRINT "STAGE "
   B-4:LOCATE 5,8:PRINT A$:LOCATE 0,12:PRINT "IN DA-
   TA <"A" MAX TIME="B
90 F=F+1:IF F>B THEN 170
95 LOCATE 0,16:PRINT "
                      "
100 LOCATE 0,16:PRINT "TIME="F " IN DATA=";:INPUT D
110 IF D<>X THEN 200
120 IF A=32767 THEN CLS:FOR I=5 TO 20:COLOR I,7,1:
    NEXT:LOCATE 5,6:PRINT "CONGRATULATIONS !!!":GO-
    SUB 250:PAUSE 10000:GOTO 180
130 CLS:FOR I=5 TO 20:COLOR I,7,1:NEXT:LOCATE 5,7:
    PRINT "YOU ARE RIGHT !!!":GOSUB 400
140 LOCATE 5,11:PRINT "PLEASE GO ON TO STAGE "B-C:
    PAUSE 10000:
150 IF A=16384 THEN A=32767:B=B+1:GOTO 70
160 A=A*2:B=B+1:GOTO 70
170 CLS:G=G+1:LOCATE 5,5:PRINT "TIME IS OVER "B"!!!":
    LOCATE 4,10:PRINT "CONT or END ? (Y/N)":Y$=INKEY
    $(0):IF Y$<>"N"AND G<4 THEN 70
```

```
180 CLS:LOCATE 8,8:PRINT "GOOD BYE!!!":LOCATE 24,20:
    PRINT "      ":END
200 LOCATE 0,20:PRINT "
                          ";GOSUB 350:LOCATE 0,20
210 IF D>X THEN PRINT D"is big! Give a smaller data":GOTO 90
220 PRINT D "is small! Give a bigger data":GOTO 90
250 PLAY "M1V12Y3T2:M0V9M1V9Y2T2:M1T2"
260 PLAY "O3E5GA:O2G5O3CC:O2C5RE"
270 PLAY "O4C6D3O3AO4C:O2GO3EC:GRE"
280 PLAY "O3G5E6G3:O2GGO3C:CRE"
290 PLAY "DCDE:O2GO3C:GR"
300 PLAY "G9:CO2GGO3C:ECGC"
310 PLAY "E5GAO4C6D3:O2GO3CCO2GO3E:CREGR"
320 PLAY "O3AO4CO3G5E6G3:CO2GGO3C:ECRE"
330 PLAY "DCDEC9:O2GO3CC9:GRECGE"
340 RETURN
350 PLAY "M1V12Y0T2:M0V9Y2T2:M1T2"
360 PLAY "O2B3ABAGA:O2C5EG:O1G5O2C3CC5"
370 PLAY "GAO3CO2AO3CR:EC7:O1G3GG7"
380 RETURN
400 PLAY "M1V12Y1T3:M0V9M1V9Y2T2:M1T2"
410 PLAY "O3G3O4CO3GO4C:O3C5E:O2G5C3C"
420 PLAY "O3AOACO3AGEG:GEC:C5G3GG5"
430 PLAY "EDCEC:O2GAG:C3CC5G3G"
440 RETURN
```

程序说明：

1. 15、20 两行打印游戏标题头。其中 15 行是使"GUESS NUMBER"显示黄色，以突出标题。

2. 25～60 行为游戏者选关处理程序，游戏者可从 1～11 关中任选一关开始游戏。从第一关至第 11 关的猜数范围依次是：0～

32、0～64、0～128、0～256、0～512、0～1024、0～2048、0～4096、0～8192、0～16384、0～32767。

3. 70～100 行的作用是在选定关次后，由电脑自动生成该关取数范围内的随机数，要求游戏者输入试猜数。

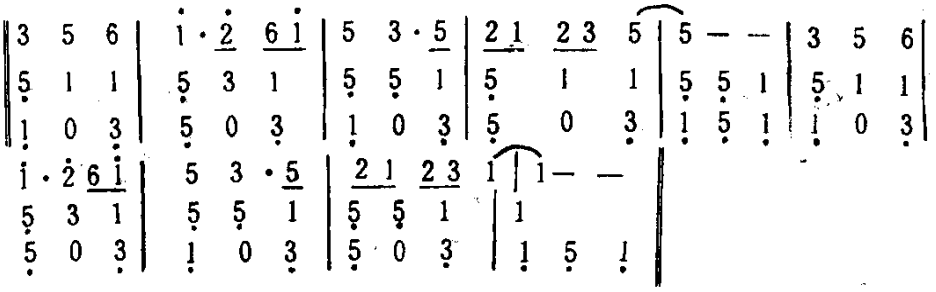
4. 110 行为游戏者没有猜中的转移处理。

5. 120～160 行为猜中的处理程序。其中 120 行是游戏者冲破 11 关的处理程序，这时打印金黄色的贺词“CONGRATULATIONS!!!”。130～160 行是升级——进入下一关的处理程序。

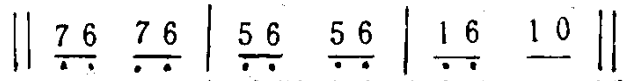
6. 170～180 行为结果处理程序。其中 170 行中记录了游戏者的接关次数，若接关 3 次仍没有猜中，则进行结束处理。

7. 200～220 行的作用是给游戏者提示猜数的方向（猜大点或猜小点）。

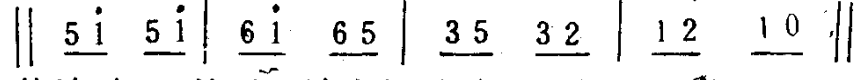
8. 250～340 行为标题、祝贺音乐子程序。其简谱为：



9. 350～380 行为游戏者猜错音乐程序。其简谱为：



10. 400～440 行为游戏者猜中音乐程序。其简谱为：



你能连破 11 关吗？请上机试试……。

结果怎么样？

你可能侥幸猜中一次，但大多数猜不中。尤其是猜数范围较大时，更难进入下一关。

那么，能否连破 11 关呢？

回答是肯定的。

7.1.3 《猜数游戏》的攻关方法

任何电视游戏都有其特定的攻关技巧，一般称为攻关密诀。《猜数游戏》也不例外，它也有其特定的攻关方法。下面以第一关为例说明实现的方法。

第一关猜数范围 0～32，且猜数小于 32。假定电脑产生的随机数是 19，第一次我们猜 16，电脑则提示“猜数小了，给一个大一点的数”；那么我们第二次在 16～32 之间选择，猜中间数 24，电脑提示“数大了，猜小一点的数”；现在的猜数范围已缩小到 16～24 之间（不包括 16 和 24），第三次仍猜中间数，选择 20，则提示：“数大了，猜小一点的数”；现在的猜数范围已缩小到 16～20，第四次在 16～20 之间选中间数，猜 18，电脑提示“数小了，猜大一点的数”；现在的猜数范围是 18～20，且不包括 18、20，显然电脑选择的数是 19，因此第五次猜 19 必然猜中。

我们可以再换一个数，例如电脑选定的随机数是 7。按上述方法必能在第五次猜中。留给读者验证。

通过以上的叙述和实践，读者可能会问：为什么最多键入 5 次以后便可在 0～32 范围内找到任意一个随机整数呢？为什么不是 4 次或者 6 次呢？又为什么恰好是 32 而不是 31 或者 33 呢？下面我们来探讨这一问题。

首先看一个等式：

$$2^5=32 \qquad (7—1)$$

式中的指数恰为我们试猜的次数，右端的 32 恰为我们的取数范围，那么底数 2 就是我们猜数时对猜数范围对半分的根据（即每次猜数均取猜数范围的中间数）。

猜数范围对半分的方法，在应用数学中叫“折半查找法”，或叫“折半搜索”。它是计算机数据查找中最常用的方法。

我们把(7—1)式改写如下：

$$Log_2 32=5 \qquad (7-2)$$

上式表示在 0～32 之间猜数最多猜 5 次。

根据式(7—2)，依次有如下等式：

$$Log_2 2=1 \qquad (7-3)$$

$$Log_2 4=2 \qquad (7-4)$$

$$Log_2 8=3 \qquad (7-5)$$

$$Log_2 16=4 \qquad (7-6)$$

式(7—3)表示在 0～2 之间猜数只要一次便可猜中。式(7—4)表示在 0～4 之间猜数最多要 2 次。很显然在 0～4 之间猜一次猜数范围就缩小为式(7—3)的形式。所以说在 0～32 之间猜数最多 5 次必能猜中。

同理有

$$Log_2 64=6$$

$$Log_2 128=7$$

$$Log_2 256=8$$

……

由程序的 50 行可以看出，猜数范围每增大一倍，猜数次数增加一次，所以，无论猜数范围有多大，都可在规定次数内猜中。因此，读者只要按“折半查找法”进行猜数，必能逢关必过，顺利突破 11 关。

7.1.4 示范游戏《爱的结合》

《爱的结合》是一个由电脑自动完成的示范游戏。游戏开始，在简短的音乐声中，屏幕上画出了红色的心形图案，随后在屏幕的左、右下角分别走出了玛莉哥哥和丽莎小姐，两人相向而行，在屏幕中心相遇。相遇后，显示大红色的底背景，奏起了欢快的祝贺音乐，打印出“爱的结合”的祝词，结合的心放射出灿烂的光芒。

程序清单如下：

```
5 REM "No. 7—2 PERFECT"
10 PLAY "M1V9Y2T3:M1V7Y1T3:M1T3"
15 PLAY "O3E5G:O3C5C:O2G5G"
20 PLAY "O4CO3B4A1:CC:#F#F"
25 PLAY "G4A1:F4F1:O3D4O2B1"
30 PLAY "B5O4C7:F5E7:G5C7"
35 CLS:ERA 0,1:SPRITE ON:CGSET 0,0:U=&H15:PALETB 0,
   15,48,48,U
40 A$="AIIIIIIIIIIIGIGGGGH":B$="HHBHBBBEBEEE":K$=
   "ABCDEFGHI":I$="ABCGHIDEF"
45 LOCATE 9,10:PRINT "START!!!":PAUSE 80:CLS:X=13:Y=
   22:I=0
50 GOSUB 100:SWAP A$,B$:GOSUB 100
55 IF I=0 THEN SWAP K$,I$:I=1:GOTO 50
60 GOTO 150
100 L=LEN(A$):FOR P=1 TO L:K=P:IF I=0 THEN 110
105 K=L+1-P
110 C=INSTR(K$,MID$(A$,K,1))-1:V=C/3:H=C MOD 3
115 X=X+H+3*(H=2):Y=Y+V+3*(V=2):LOCATE X,Y:
    PRINT CHR$(207);:NEXT:RETURN
150 W=&H34:FOR N=0 TO 1:DEF MOVE(N)=SPRITE(N,3+4
    *N,1,1,0,0):POSITION N,236*N,200:NEXT
155 MOVE 0,1:IF CRASH(0)=1 THEN 165
160 GOTO 155
165 LOCATE 10,10:PRINT "PERFECT":FOR N=0 TO 100:
    PALETB 0,6,&H30,&H30,U:SWAP U,W:NEXT
170 PLAY "M1V2Y1T2:M1V10Y1T2:T2"
175 PLAY "O3E3G#DGDG#DG:O1#B5B#AB:R"
180 PLAY "EGO4CO3E:Y2O2#B6E3:O1C5C"
185 PLAY "#GAO4DC:F5#F:#C#C"
190 PLAY "O3B#ABO4A:F3O3DFA:D6F3"
```

```
195 PLAY "GO3AO4FO3B:E5DV15:GA#AB"
200 PLAY "#B3GT3FGE1DV12E7:C9:#B5T3GC7"
210 LOCATE 0,22:END
```

程序说明：

1. 10～30 行为开始音乐。

2. 40～115 行打印心形图案，其原理已在第五章 5·2·7 中例 5·10 中作过分析，不再赘述。

3. 150～160 行为定义卡通图案运动程序。

4. 165 行的作用是使图案闪光。

5. 170～200 行是祝贺音乐程序。

通过阅读分析以上两个简单的游戏程序，我们得出结论如下：游戏程序是具有某种游戏功能的特殊程序。它同其它任何程序一样，具有输入、输出、循环、判断、结束等处理语句，所不同的是，它比一般程序结构更复杂、内容更丰富。所以，我们说：

游戏程序是电脑实力的充分发挥，是电脑计算、判断、声音、图象、动画等各种功能的综合表现。

7.2 故事情节和游戏结构

电视游戏有许多种类，一般分为：战争、冒险、管理、智能四大类。不管哪一类游戏，都有一个中心内容。它们都能够完全细致地描述一个事件的发生、发展、结束全过程；都有一定的知识性、启迪性或趣味性；都有一定的难度和技巧。以激发游戏者的好胜心和竞争意识，促使他们去思索、推理、判断、总结，从而达到游戏中增长知识，娱乐中提高才干的目的。

那么，游戏程序是如何编制的呢？它一般要经过确定故事情节、规划游戏结构、设计程序结构、绘制程序框图、编写详细程序五个步骤。如果把游戏比作机械产品，那么，故事情节就是原材料；游

戏结构就是加工图纸；程序结构就是零件毛坯；程序框图就是工艺文件；编制程序就是产品的最后加工。

7.2.1 故事情节的概念

众所周知、机械图纸是机械行业加式制造的依据。游戏结构则是编制游戏程序的关键，是故事情节的再创作。那么，游戏结构的内容如何呢？欲说清这一问题，还须从故事情节开始。

我们知道许多战争、冒险类游戏都是从一个特定的故事情节开始的。如最为游戏者所喜爱的游戏《魂斗罗》的故事情节是这样的：公元 2631 年，世界进入了和平时期，地球海军陆战队司令部在绝密调查中发现了异星侵略者消灭人类的计划。海军陆战队所属上等兵比尔·拉依乍和兰士·比恩接受了歼灭异星侵略者的命令。于是二人便向已变成异星侵略者要塞的嘎尔嘎各岛进军。他们要实破原始森林、外围基地、瀑布天险、防御基地、雪地屏障、动力中心、飞机仓库七个关口，最后抵达外星人栖息处，击毙外星人，摧毁敌基地后，乘飞机胜利返航。

7.2.2 游戏结构的概念

故事情节犹如文学创作的素材，有了故事情节才能决定游戏结构。如《魂斗罗》的游戏结构是这样的："这是一个战争类游戏，因此首先给主人翁配备了光电子弹、火焰喷射器、火球、流星、光电幅射弹等各种武器；其次，由于要经过各种复杂的地带，主人翁要具有潜水、飞腾、跳跃、射击的本领；第三要设计各种不同的游戏画面、音乐及与其对抗的敌人，以加强气氛、增加真实感；第四为增加游戏难度，把整个游戏分成了八大关，每关的关头都要与一个势力较强的敌人搏斗，若战术不力很难过关等等"。

归纳起来，所有游戏的游戏结构本质上都是相同的，它一般包含以下几项内容：

1. 游戏的背景画面

游戏的背景画面如同电影、戏剧中的布景，使用得当，可极大

地增强游戏的艺术效果，使游戏更形象、更生动、更富有吸引力。一般要根据游戏的故事情节，决定画面的种类、色彩、数量、变化方式和复杂程度。

2. 游戏的音响效果

为使游戏逼真地描述再现故事情节，一般游戏中都配有一定的效果音响。如开枪、炸弹爆炸、敌人被击中、建筑物被摧毁等情节，都要有特定的音响，以烘托游戏的气氛。再如飞机、汽车的引擎声、人和动物行走的脚步声等，都起到了增强游戏的真实感的作用。所以说，精彩的游戏与丰富的音响是密切相关的。

3. 文字说明

文字说明一般包括故事梗概介绍、游戏玩法说明、人物对话和时间、积分、奖励、游戏机会、关次、游戏者等方面的提示、叙述和表达，以弥补画面显示的不足。

4. 游戏者的动作

所谓游戏者是指游戏中的主人翁（可能是战士、拳王、历史名将、巾帼英雄，也可能是猪、狗、猫、兔、老鼠、妖怪等动物）。游戏中要规定主人翁的动作和本领。如跑、跳、潜水、腾空、射击、驾驶等。主人翁每一次移动后，计算机都要计算其移动后的结果，若主人翁已经取胜或失败，则进行结束处理（或者重新开始、或者接关继续）；否则，主人翁或另一位对手作另一次移动。

5. 游戏的选项

一般游戏中都会有选项操作，如选择关次、人数、游戏难度等。选项的类型、选择的方法和变量的修改，一般要根据我们所选中的故事情节确定。一个大的选项可能衍生出多层选项。如我们选中“攻击”，则随后就要有“射击、出拳、踢腿”等动作的选项。若选中“射击”，则又有各种武器和弹药数量的选项等等。选项一般与文字说明共存，也可以用画面选项或者画面、文字混合选项。每个游戏选项的多少、选项的层次与游戏的结构有关，同时还受计算机内存

容量的约束限制。

6.游戏的难度

一个有趣的游戏既要容易理解又要有足够的难度，既要容易入门又要在短时间内很难玩好。设置难度惯用的方法是设计很多关(如《魂斗罗》有八大关，《超级玛莉》则有八大关三十二小关，而《淘金者》则有50关。)，开始几关较容易攻破，以后难度越来越大，要求技巧越来越高。只有这样，才能促使游戏者动手、动脑、去思索、去探讨，津津有味的玩下去。如多版游戏《淘金者》有Ⅰ、Ⅱ两代，每代各有50关。第一代的50关较容易攻破，第二代的50关难度比较大，而且关次越高难度越大。有些关次要求动脑，有些关次要求技巧，如第Ⅱ代的第19关、31关就较难突破。尤其是第50关要求动脑、技巧共用，难度最大。必须进行精心的策划；合理的安排顺序；快速、准确的行动才能顺利过关。因此《淘金者》是一个“智能”与“技能”密切结合的典型游戏。另外，有的游戏难度的设置是通过改变敌人的数量、敌人的生命力、缩短限定时间等方法实现的。如《双截龙》游戏就有“容易、一般、困难”三种难度选择。选择“容易”时，一个“冲天拳”就可以打死敌人，而选择“困难”时就要打三次才能把敌人打死。只有这样，才能刺激游戏者的好胜心，提高游戏者的兴趣。不难想象，一个很容易取胜的游戏，肯定会令人乏味的。

游戏结构可粗略地划分为以上6点，但并非所有的游戏都要具备这6点。例如智能类、管理类游戏，只要有1、2、3、5、6、几项就可以了，甚至可以更少，只有3、6项也可以构成游戏。但不具备以上几点内容之一的程序，肯定不是游戏程序，既使勉强凑成，也不会引起人们的兴趣。

另外，在规划游戏结构时，还须兼顾电脑系统提供用户使用的内存余量。象《魂斗罗》类的战争游戏，结构相当复杂，程序极为庞大，用机器语言编写，占用内存256K。而学习机，一般只有4～8K

的内存供用户使用，所以我们只能编一些简单的游戏程序。尽管如此，游戏结构也是必不可少的。它如同建筑中的房架结构，没有房架肯定造不成房屋。因此，在编制游戏程序前，要精心规划好游戏结构，以使较少的内存发挥较大的作用，利用有限的内存空间编制出富有趣味的游戏程序。

7.2.3 故事情节示例

现在，我们构思一个这样的游戏，名字叫做《成龙救金凤》。故事情节如下：

“成龙和金凤是一对热恋的情侣。山妖发现金凤的美貌，就派秃鹰叼走了金凤。欲打死山妖，救出金凤，必须找到克制山妖的宝贝——降魔镖。但是降魔镖是山神的镇山之宝，从不借人，只有觅机盗出。降魔镖秘藏在聚宝楼内，并有两名妖怪巡回看守，而聚宝楼是一座全封闭的楼房，每月只开一个时辰通风，成龙只有在通风时进入楼内，躲过妖怪，寻到宝贝，在通风时间之内离开聚宝楼。否则，将葬身聚宝楼。若被妖怪抓住也立即毙命。勇敢的成龙，抓住时机，机智地躲过妖怪，终于寻到宝贝，打死山妖，救出爱侣。”

7.2.4 游戏结构示例

根据故事情节，我们规划“成龙救金凤”游戏结构如下：

1. 游戏画面

游戏画面共设计四幅：开始画面一幅；主题画面一幅；结束画面两幅。

(1)开始画面

成龙和金凤沐浴着落日的余辉，漫游在郊外小径。突然，秃鹰自天而降，叼起金凤，腾空而去。成龙转身狂奔急追。

(2)主题画面

一座七层楼房。金凤显示在楼顶右上角。成龙在楼下左下角。两个妖怪随机出现在3～7层楼之间。九个大小、形状各异的藏宝箱放置在七层楼的不同角落。

(3)结束画面

一幅是成龙、金凤二人举手欢呼、相向奔跑、热烈拥抱。一幅是成龙、金凤二人迈着轻快的脚步，手拉手高高兴兴地回家。

2. 游戏音响

(1)游戏开始有一段悠扬的乐曲，伴随着成龙和金凤愉快的散步。突然天空变暗，秃鹰降临，叼走金凤，成龙狂追。这时乐曲变得紧张、急促、强烈。

(2)成龙救出金凤，相向奔跑、热烈拥抱时，有一段激动欢快的音乐。

(3)成龙与金凤手拉手回家时，又是一段悠扬的音乐伴随着他们欢快的脚步。

(4)游戏的进行中，穿插一些简短的象声音响，如脚步声、上楼、下楼声、打开箱子的声音等等。

3. 文字说明

游戏比较简单，不作说明。只打印题头、时间、得分等。

4. 动作

该游戏的主要情节是成龙依次打开箱子，寻找宝贝。因此，成龙的动作仅设计走路、上楼、下楼三个简单的动作。可用左、右键控制成龙左、右走，用上、下键控制成龙上、下楼梯，用"START"键控制游戏开始。

5. 游戏的难度

(1)为增加游戏难度，从 27 个图形中随机选取 9 个作为藏宝的箱子。箱子有一定的顺序(由电脑自己约定，不标出)，成龙只有按顺序打开第一个箱子，然后再打开第二个、第三个……，直到打开第九个。这犹如宝贝放在第九号箱子里，九号箱子的钥匙放在八号箱子里，而八号箱子的钥匙又放在七号箱子……。所以必须从第一号箱子开始，依次打开，才能最终找到宝贝。

(2)两个妖怪在七层楼之间穿梭巡逻，并自动向成龙靠近，以

干扰成龙的寻宝速度。

(3)增加时间限制,成龙必须在规定时间内依次打开九个箱子,若超过规定时间,没有完全打开,则本次游戏结束。

为压缩程序量,本游戏省去选项。欲增加游戏难度,可缩短时间或增加巡逻妖怪的数量。

以上就是"成龙救金凤"游戏的游戏结构。规划好游戏结构以后,就可进入下一步骤了。

7.2.5 游戏介绍——《石头、剪刀、纸》

例 7.2 赌博游戏——《石头、剪刀、纸》

石头、剪刀、纸的游戏(也叫剪子、包袱、锤)是小朋友们最常玩的赌胜游戏。游戏中,双方在石头(Rock)、剪刀(Scissors)、纸(Tissue)三者之间任选其一。规则是:石头击败剪刀,剪刀击败纸,纸又击败石头。为了取胜,游戏者必须仔细研究对方的战术,随时调整自己的选择以击败对方。该游戏虽然结构简单,但却饶有趣味。

这个游戏,我们也可以与电脑一起玩。程序清单如下:

```
5 REM "No. 7—3 GAMBLE GAME"
10 CLS:CLEAR:CGSET 0,0:ON ERROR GOTO 260
20 LOCATE 7,3:PRINT "* GAMBLE GAME *"
30 LOCATE 0,9:PRINT "GUESS ROCK, SCISSORS, TISSUE":
   PRINT
40 PRINT " BY PRESSING〈R〉,〈S〉,〈T〉":PRINT
50 PRINT " TO END GAME PRESS〈ESC〉":PRINT
55 LOCATE 10,18:PRINT "1992. 3."
60 GOSUB 360:CLS:GOSUB 320
70 PLATEB 0,RND(12)+1,13,29,45
75 LOCATE 0,0:PRINT "YOUR TOTAL ";T(2)
80 LOCATE 15,0:PRINT "MY TOTAL ";T(1)
90 LOCATE 6,8:PRINT "                    ":LOCATE 18,8:
   PRINT "                    ":LOCATE 10,15:PRINT "
```

```
                    "
100 M=1+ABS(N(S,1)<=N(S,2))
110 M=M+ABS(N(S,M)<=N(S,3))*(3-M)
120 R=S:M=M+3*ABS(M=1)-1
130 LOCATE 0,8:PRINT "YOUR BET   ";:G$=INKEY$(0)
140 IF G$="R"OR G$="S"OR G$="T"THEN 170
150 IF ASC(G$)=27 THEN 260
160 GOTO 130
170 X=0:GOSUB 290:PRINT MID$(P$,X,9)
180 S=ASC(G$)-81:N(R,S)=N(R,S)+1
190 LOCATE 15,8:PRINT"MY BET   ";:G$=CHR$(M+81)
200 X=0:GOSUB 290:PRINT MID$(PS,X,9)
230 IF S=M THEN LOCATE 10,15:PRINT" A DRAW":GOSUB
    360:GOTO 90
240 W=S-M+ABS(M>S)*3:T(W)=T(W)+1
250 LOCATE 10,15:PRINT N$(W)" WIN":GOSUB 360:GOTO 70
260 CLS:PRINT:IF T(1)>=T(2) THEN PRINT "YOU lose!":GO-
    TO 280
270 PRINT "You are very clever!"
280 LOCATE 0,20:END
290 FOR I=1 TO 24 STEP 9:IF G$=MID$(P$,I,1) THEN X=I
300 NEXT:RETURN
320 P$="ROCK          SCISSORS TISSUE      "
330 DIM T(2),N(3,3)N$(2)
340 R=RND(3)+1:S=RND(3)+1
350 N(R,S)=1:N(S,R)=1:N$(1)="I":N$(2)="YOU":RE-
    TURN
360 LOCATE 0,22:PRINT "Prees space bar to continue"
370 A$=INKEY$(0):IF A$<>" " THEN 370
380 LOCATE 0,22:PRINT "
                    "
```

390 RETURN

程序说明：

1. 10～70 行打印游戏标题和游戏说明，并随机改变背景的颜色，以使屏幕显示灵活多变。

2. 75～80 行打印游戏者和电脑各自的得分。

3. 90 行的作用是清除上次打印的游戏者，电脑所选的内容及胜负情况的评判结果。

4. 100～120 行的作用是计算并确定电脑本次所选的内容。它是根据以前游戏者选择的规律而计算出来的。这三行是本程序的关键，它决定了电脑的对策。

5. 130～170 行的程序记录了游戏者所选的内容，并打印在屏幕上。其中 150 行为结束转移程序。

6. 180 行记录游戏者本次所选的内容。

7. 190～200 行打印电脑所选的内容。

8. 230～250 行的作用是判定胜负、累计得分。其中 230 行是对两者所选内容相同时的处理；240 行对胜的一方加一分；250 行打印哪一方胜。

9，260～280 行是结束处理程序。若电脑积分多，则打印“你输了！”；若游戏者得分多，则打印“你真聪明！”。

10. 290～300 行是确定两方所选内容名字的全称的子程序。因在选择输入中，只输入 R、S 或 T，则由该子程序翻译出三种选项的全称，并打印出来。

11. 320～350 行是初始赋值子程序，也可以把它放入主程序的 60 行之后。

12. 360～390 行是场次转换子程序，每进行一场比赛，按空格键，再进行下一轮比赛。

现在，我们来玩玩这个游戏。

运行程序，清屏。显示图 7—4 画面：

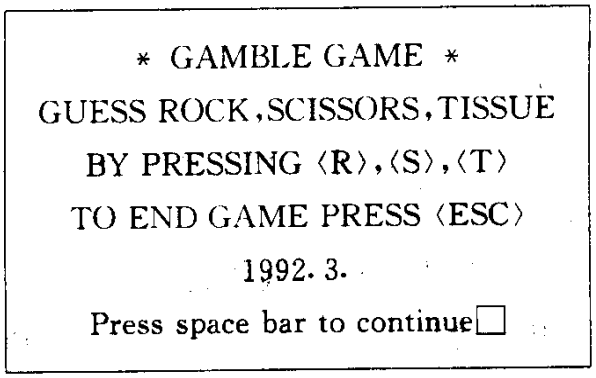


图 7—4　例 7.2 开始画面

图 7—4 的内容是:“赌博游戏”。“猜石头,剪刀,纸”。“分别按〈R〉、〈S〉、〈T〉键”。“如果结束游戏则按〈ESC〉键”。“按空格键开始游戏”。

按空格键后,清屏。显示如图 7—5

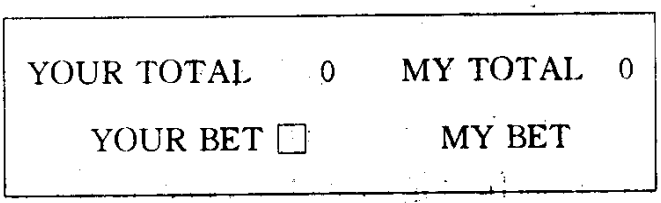


图 7—5

图 7—5 第一行是:左边“你的全部得分”(指游戏者);右边“我的全部得分”(指计算机)。第二行的意思“你打赌什么”,闪烁光标等待你按〈R〉、〈S〉、〈T〉三键中的任一键。例如我们选石头,则键入 R。假如计算机选择剪刀,则打印图 7—6 画面。

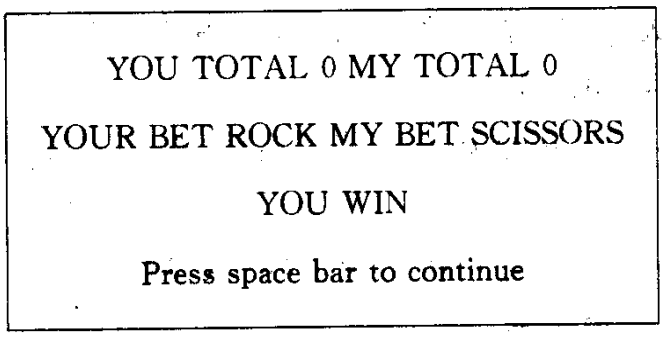


图 7—6

图 7—6 的意思是:“你赌的是石头,我赌的是剪刀”。“你赢

了”。“按空格键继续游戏”。

若按空格键，则在图 7—6 中的第一行左端的“YOU TOTAL 0”变为“YOU TOTAL 1”。表示你现在的全部得分为 1 分。同时图 7－6 变成图 7—5 的画面，等待你的第二次选择输入。假如第二次我们选剪刀，电脑也选剪刀，则显示图 7—6 形式的画面。所不同的是第二行的左端为“YOUR SCISSORS”，在第三行“YOU WIN”的位置显示的是“a DRAW”，意思是“平局或不分胜负”。“按空格键继续游戏”……。

若欲结束游戏，则按〈ESC〉键。计算机将根据双方得分的多少打印出评语。或者为：

“You are very clever!”

“你真聪明!”表示你的积分高于电脑的积分。若电脑的积分高则打印：

“You lose!”

“你输了!”。

读者可以多玩几次，试一试能否战胜计算机。

通过实践，可以发现，如果我们以固定的规律玩这个游戏（即选完石头选纸、选过纸后选剪刀等等），三、五轮以后，得胜的一方总是计算机。只有不停的变换选择规律，才有可能战胜电脑而赢得高分。

7.2.6 赌博游戏中电脑决策的分析

在例 7.2 游戏中，为什么电脑会自动改变选择？又为什么它的选择正是战胜游戏者的选择？电脑选择的决策依据是什么？……。现在我们分析讨论这一问题。

首先，我们看一张表（表 7—1）：

表 7—1　游戏者选择统计表

		紧跟着的选择		
		石　头	剪　刀	纸
前次选择	石　头	0	8	1
	剪　刀	9	1	1
	纸	0	3	9

表 7—1 是一个 3×3 阵列的统计表。表中的每一个单元与程序中 N(3,3)数组的一个下标变量对应。电脑把游戏者以往的选择都记录在这张表中。轮到电脑选择时，它首先查看表 7—1，看一看游戏者上一次选的是什么，习惯于哪种规律的选择，从而做出克制游戏者的选择决策。例如，在表 7—1 第一行中，游戏者在前次选完石头后，接着选石头的次数是 0 次；接着选剪刀的次数是 8 次；接着选纸的次数是 1 次。显而易见，游戏者选过石头后接着选剪刀的可能性最大。因此，当游戏者选过石头后，电脑根据表 7—1 可判断出游戏者紧接着要选剪刀，所以它做出了克制剪刀的选择，选定石头。游戏中，电脑一直盯着这张表，以做出取胜的选择。这就是电脑决策的依据。

当然，在实际游戏中，游戏者一旦意识到电脑已经掌握了他的规律，就会选用另外的策略。而当游戏者采用的策略使用几次后，电脑又会相应地选择对付游戏者的策略。不难想象，如果游戏者固定采用一个惯用策略，那么得胜一方将总是计算机。实际上，绝大多数游戏者都会意识到这一点，及时改变策略，以谋取胜。

至于电脑如何填写“游戏者选择统计表”，在程序中，主要体现在 100、110、120、180 四行程序，读者可自行分析。

7.3 程序结构和程序框图

7.3.1 程序结构

程序结构一般分为三大类：

· 模块结构

· 顺序结构

· 混合结构

1. 模块结构

模块结构也叫结构化程序。它根据总体需要和系统配置的可能，把一个程序分成具有某些特定功能的程序段，每个程序段形成一个子程序，称为功能模块。每个模块相对独立地完成一定的任务：它可以是计算分析战斗场面；可以是游戏主人翁的某些动作；也可以是一段音乐伴奏等等。然后再加一个调用各子程序的主程序，称为主控程序或管理程序。管理程序在需要时可调用所有模块。另外，各模块之间也可以互相调用。不难看出，一个完整的程序就是由管理程序和若干模块组成的。这种结构的优点是：①由于模块之间相对独立，所以在开发程序时，可以分头编写，程序的调试、改错较容易，因而，编程周期短；②由于模块的相对独立性，能有效地防止错误在程序中的扩散与蔓延，因而提高了系统的可靠性和质量；③由于各模块功能明确，相对独立，通用性强，因此便于其它程序的借用和向其它语言的移植。缺点是：编程时要加一定的限制；调用时要考虑有关变量的赋值；调用后要对运算结果进行恰当的处理。因此，结构化程序的程序量较大。一般在大型、复杂程序中使用。

2. 顺序结构

顺序结构也叫非结构化程序，它与结构化程序相反，采用顺序编写的方法。程序中的转向，使用 GOTO 语句实现，条件转移语句

用得较多。这种结构的优点是:①程序顺序编出,变量之间不易混淆;②省去了程序的链接、调用操作,从而程序量较小;③由于GOTO、IF——THEN 语句使用较多,程序不易阅读,从而使其保密性较好。缺点是:只能一个人编写程序,编程周期长;程序的调试改错较困难,一处有错,殃及整个程序不能运行;由于出现 GOTO 网络,不易阅读理解,故借用移植性较差。这种结构一般用于简单程序中。

3. 混合结构

混合结构则综合了以上两种结构的优点。它根据游戏的需要,对模块化结构程序进行了灵活处理。把那些整个游戏中只调用一次的模块则取消模块格式,直接放入主程序中;对调用最频繁的模块,则尽量安置在行号较小的位置,甚至移到主程序之前,以提高程序的执行速度(由于计算机在执行 GOTO、GOSUB 语句时,总是从最小行号开始搜索,直到目标行号)。

今后我们介绍的程序,多是混合结构的程序。但是,为使层次清楚,便于阅读理解,开始介绍时,还是以模块化结构为主。

7.3.2 程序框图

我们知道,程序框图是用各种几何图形及文字说明来直观地描述计算机计算执行过程的有向图,也叫流程图。程序框图一般分为两大类:

1. 粗结构框图

粗结构框图也叫功能模块结构图。它描述整个软件系统由哪些模块组成,也说明了模块之间的调用关系。一般有下述形式(见图 7—7):

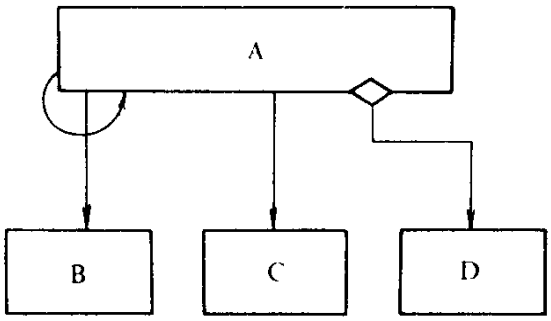


图 7—7　功能模块结构图

图 7—7 中，方框表示模块，块内写有模块的名称（有时还要写出模块的主要功能）；箭头表示起点模块对箭头所指模块的调用；小菱形符号表示有条件调用；弧形箭头表示循环调用。图 7—7 说明：A 模块循环调用 B 模块；直接调用 C 模块；有条件地调用 D 模块。

2. 细结构框图

细结构框图也叫程序框图，俗称流程图。它是对程序的精细描述，它给出模块功能的具体实现过程。它主要包括：模块内部采用的计算方法，数据的存储结构和输入、输出等方面的设计步骤。第六章中介绍的图 6—2、图 6—3 等都属于这种结构的框图。

3. 粗、细结构框图的区别

粗、细结构框图的差别是明显的，我们以例 7.2 为例具体说明它们之间的差别。

根据例 7.2 的游戏结构，我们可以画出功能模块结构图如图 7—8。

程序 No. 7—3 是一个典型的混合结构程序。

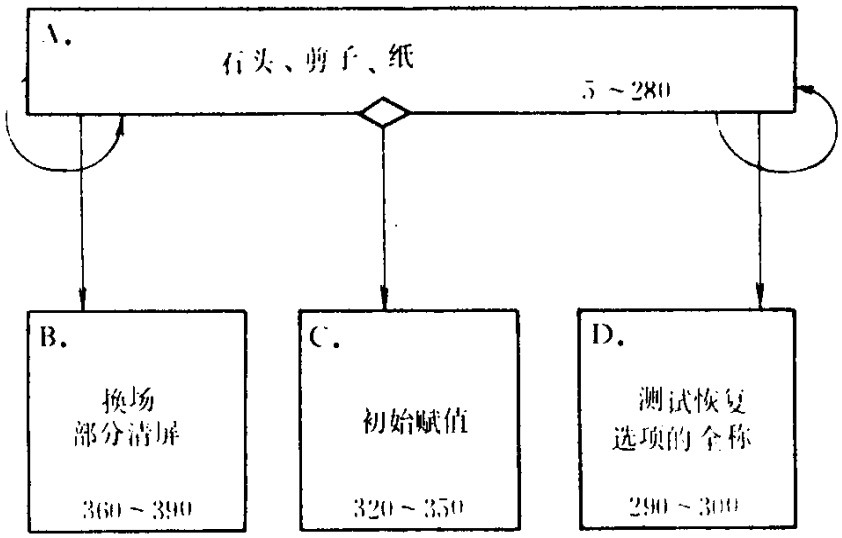


图 7—8　石头、剪刀、纸模块结构图

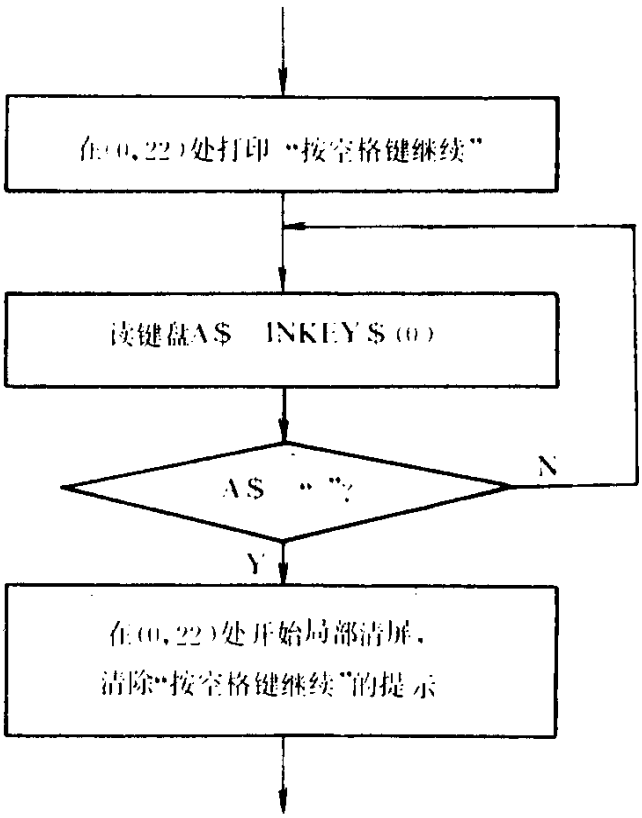


图 7—9　石头、剪刀、纸模块 B 流程图

A 为管理程序(主控程序)模块。

B、D 为循环调用模块。

C 为条件调用模块。仅在每次游戏开始时调用。

图 7—8 中换场模块 B 的流程图如图 7—9 所示。

对比图 7—8、7—9,可见模块结构图与流程图是不同的。模块结构图反映的是程序的层次特性,即某个模块要调用哪些模块,哪些模块又调用什么模块;而流程图所反映的则是程序执行的过程特性,即先执行哪一部分,再执行哪一部分。两者是全局和局部的关系。

7.3.3 编程示例——《企鹅看外婆》

在上两节的示例中,介绍了游戏程序中加入音响的方法、游戏背景及卡通图案色彩的变换技巧和键盘输入的处理。本节再介绍如何使用操纵器控制卡通运动,卡通运动越界的判断处理及卡通随机连续运动的处理方法。

例 7.3 《企鹅看外婆》

1. 故事情节

《企鹅看外婆》游戏故事情节如下:

"企鹅太太带着五个孩子去看外婆。去外婆家要穿过一条交通繁忙的马路,马路上车辆奔流不息。她只能瞅准车辆的运行空隙,一次抱一个小企鹅送到马路对面。往返五次,把孩子们都送过马路后,才能去外婆家。"

2. 游戏结构

根据故事情节,我们可以这样规划游戏结构:游戏画面是一条很宽的横向马路;马路上有七辆汽车在不同的位置,以不同的速度运行;在马路的中段画出一条较窄的纵向人行道;人行道的北端(屏幕上方)站着五个小企鹅;企鹅太太在人行道的南端(屏幕下方);选 1[#] 操纵器控制企鹅太太前、后、左、右运动;当企鹅太太运动到人行道北端时,抱起一个小企鹅,北端的小企鹅应减少一个;

当企鹅太太返回南端后，则在人行道南端应多出一个小企鹅；五个小企鹅全部转移到人行道南端则游戏结束；转移中，若企鹅太太与汽车相撞则游戏重新开始。游戏过程中还加入了企鹅太太运动的脚步声；抱起企鹅和放下企鹅的声音；把企鹅全部转移到马路对面的祝贺声以及企鹅与汽车相撞的声音共五种简单音响。为增加游戏难度，还加入了时间限制。

3. 划分功能模块，画出模块结构图

根据《企鹅看外婆》游戏结构共划分七个功能模块，各模块功能分配如下：

(1)模块 A：管理程序模块。由它完成游戏过程的控制。

(2)模块 B：初始化模块。由该模块画出马路、人行道。马路为横向，宽 12、长 27；人行道位于马路中间，为纵向，宽 4、长 14。同时，在人行道南端显示企鹅太太；在人行道北端并排显示五个小企鹅；有七辆汽车从右向左行驶，车的颜色和速度各不相同。画面见图 7—10：

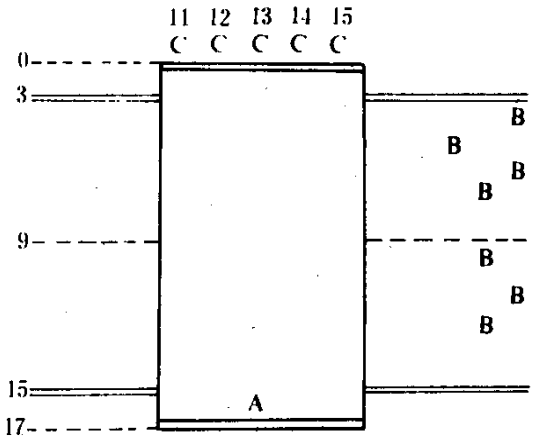


图 7—10 《企鹅看外婆》游戏画面

图中：A 为企鹅太太

B 为七辆汽车

C 为五个小企鹅

显示座标为:A:(120,144)

B:[250,RND(60)+56]

C:(104,20)～(136,20)

(3)模块 C:为汽车运行再启动模块。它随时检测七辆汽车的运行情况,发现哪辆汽车运行停止,则立即启动,令其重新运行,从而保证画面上的七辆汽车连续不断的运动。

(4)模块 D:用操纵器控制企鹅前、后、左、右移动,并随着企鹅的移动发出脚步声。

(5)模块 E:判断汽车是否与企鹅相撞,若相撞,则发声,转结束处理。

(6)模块 F:完成小企鹅的转移工作,抱起和放下小企鹅时要伴有相应的音响。若五个小企鹅全部转移完,则奏乐祝贺,转结束处理。

(7)模块 G:结束处理。用"START"钮控制游戏继续;用"A、B"钮控制游戏结束。若按"A、B"钮,则打印"GOOD--BY!",停机。

(8)模块 D 还要循环调用模块 E、模块 F。根据划分的功能模块,画出模块结构图(见图 7—11)。

4. 设计程序框图

对照各模块功能,分别画出各模块的流程图,见图 7--12～图 7--16。

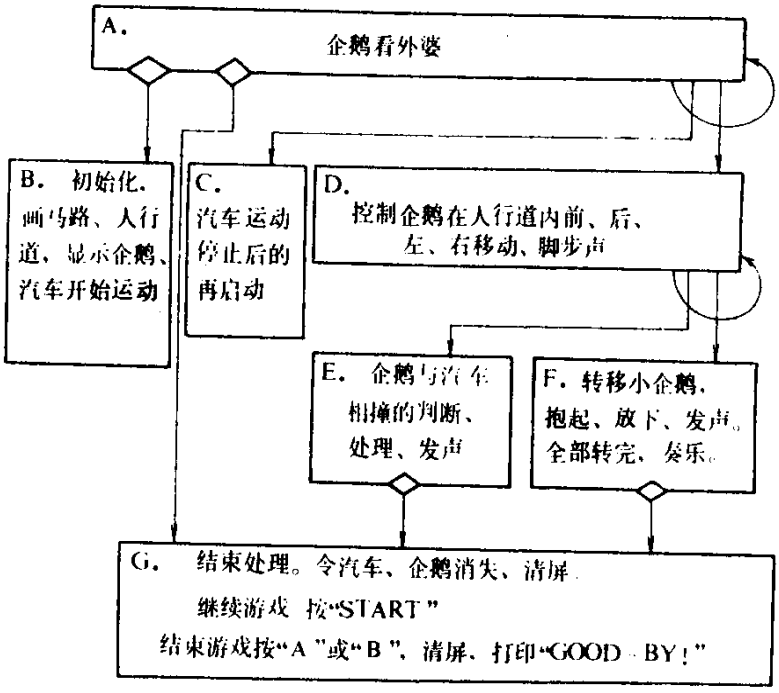


图 7—11 《企鹅看外婆》模块结构图

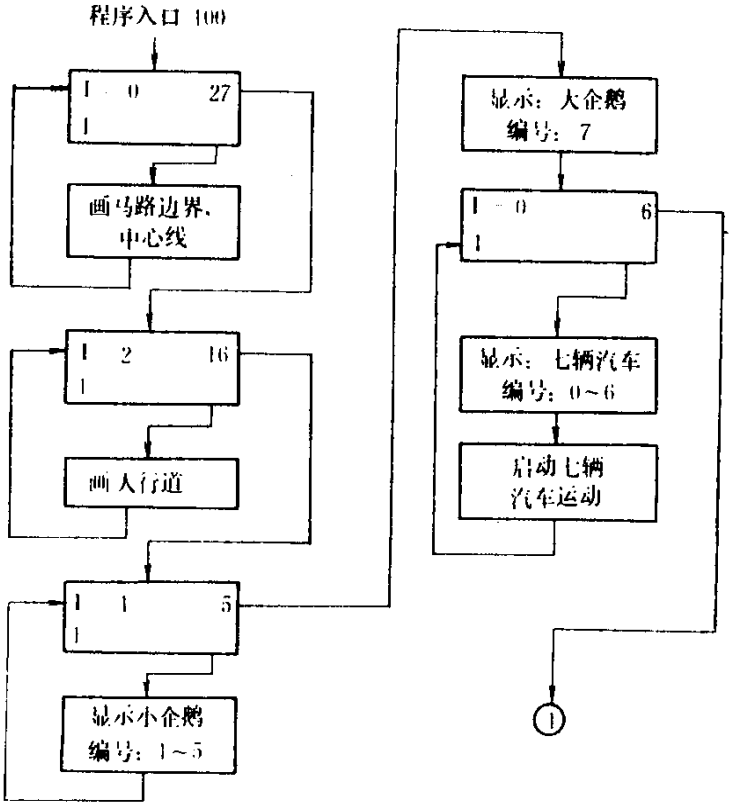


图 7—12 模块 B 程序框图

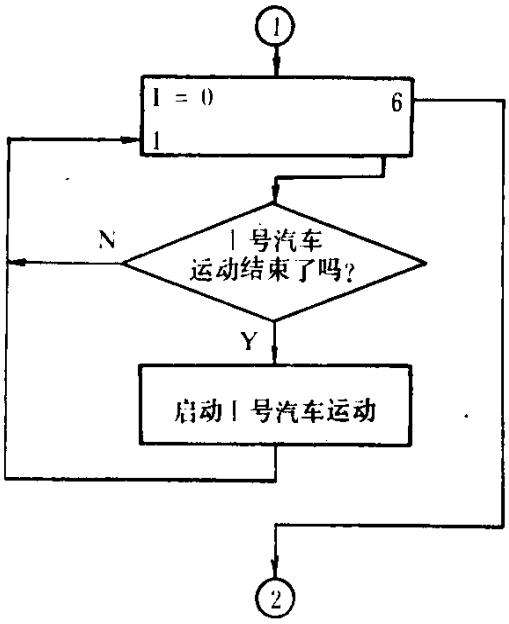


图 7—13　模块 C 程序框图

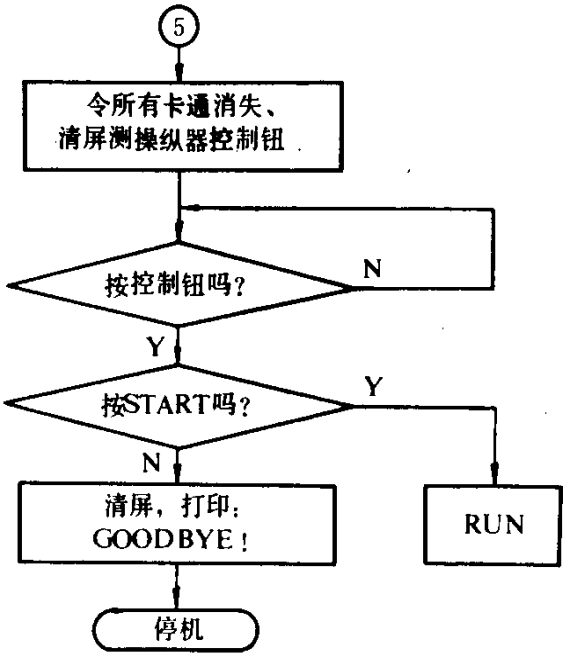


图 7—15　模块 G 的程序框图

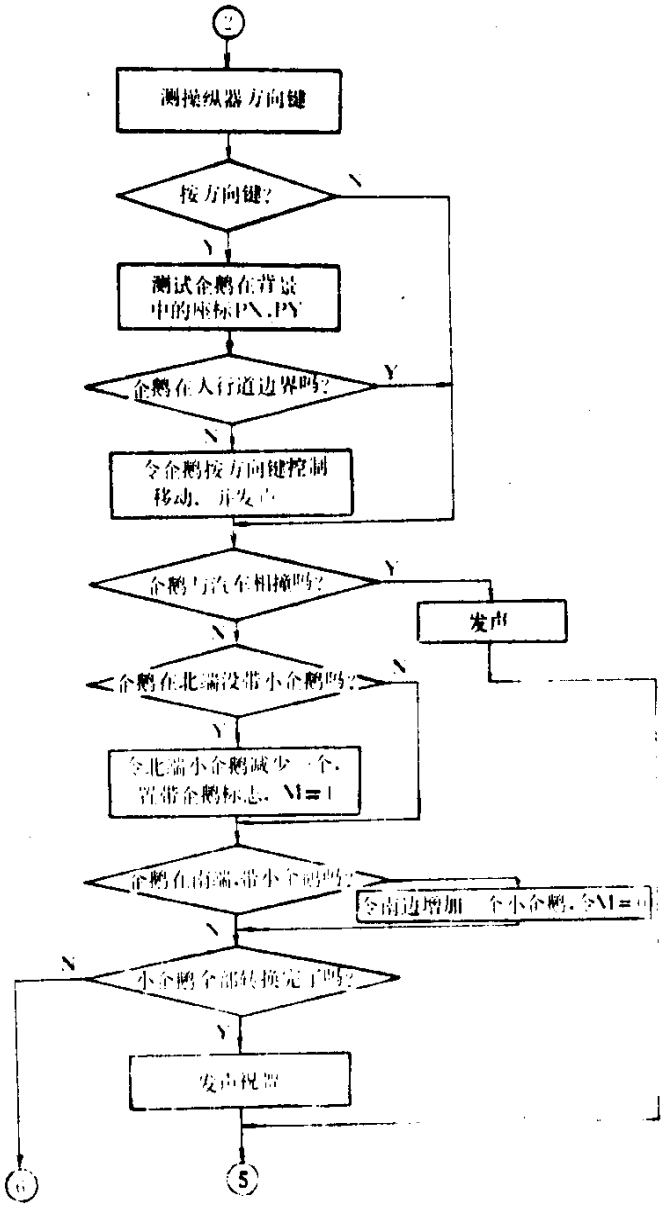


图 7—14　模块 D、E、F 程序框图

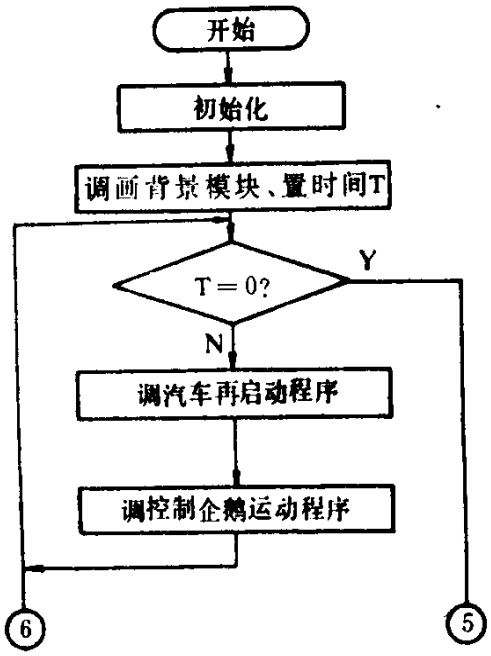


图 7—16　管理程序模块 A 的程序框图

5. 编写详细程序

根据程序框图可对应编出每个模块的详细程序，然后把各模块的程序安排适当的行号合并后，就得出《企鹅看外婆》游戏的完整程序。

该程序较简单，现直接给出结构化程序清单。(为便于阅读，特对有关语句行加入中文说明。)

```
5 REM "No. 7—4 PENGUINS VISIT GRANDMOTHER"(企鹅看外
  婆)
10 CLS:CLEAR:SPRITE ON:CGSET RND(2),0:T=800
15 PLAY "V15Y2T1:V15T1"
20 GOSUB 100(画马路,显示汽车、企鹅、小企鹅)
25 T=T-1:LOCATE 22,21:PRINT T;"    ":IF T=0 THEN 60
30 GOSUB 200(汽车的再启动)
40 GOSUB 250(控制企鹅运动)
50 GOTO 25
60 ERA 0,1,2,3,4,5,6,7:FOR I=1 TO 5:SPRITE I:NEXT
```

```
65 CLS:LOCATE 0,20:PRINT "Press START to continue"
70 S=STRIG(0):IF S=0 GOTO 70
75 IF S=1 THEN RUN
80 CLS:LOCATE 8,8:PRINT "GOOD BYE!!!":LOCATE 0,20
85 END
100 REM "Picture"(游戏画面)
110 FOR I=0 TO 27:IF I<10 OR I>16 THEN LOCATE I,9:PRINT
    CHR$(227)
120 P=3:Q=15:IF I>=10 AND I<=16 THEN P=1:Q=17
130 LOCATE I,P:PRINT CHR$(197):LOCATE I,Q:PRINT CHR
    $(197)
135 NEXT
140 FOR I=2 TO 16:LOCATE 10,I:PRINT CHR$(226):LOCATE
    16,I:PRINT CHR$(226):NEXT
150 FOR I=1 TO 5:DEF SPRITE I,(RND(4),0,0,0,0)="defg":
    SPRITE I,(I-1)*8+104,20:NEXT
160 DEF MOVE(7)=SPRITE(4,0,1,2,0,2):POSITION 7,120,144:
    MOVE 7
170 FOR I=0 TO 6:DEF MOVE(I)=SPRITE(6,7,I+1,130,0,RND
    (4))
180 POSITION I,250,RND(60)+56:MOVE I:NEXT
190 RETURN
200 REM "CARS RESTART"(汽车的再启动)
210 FOR I=0 TO 6
220 IF MOVE(I)=0 THEN POSITION I,250,RND(60)+56:MOVE I
230 NEXT
240 RETURN
250 REM "Penguin Movement"(企鹅运动)
260 S=0:SX=0:SY=0:K=STICK(0):IF K=0 THEN 380
270 IF K=1 THEN S=3:SX=1
280 IF K=2 THEN S=7:SX=-1
```

```
290 IF K=4 THEN S=5:SY=2
300 IF K=8 THEN S=1:SY=1
310 PX=(XPOS(7)-16)/8:IF PX=10 THEN PX=11
320 IF PX=15 THEN PX=14
330 PY=(YPOS(7)-24)/8
340 S$=SCR$(PX+SX,PY+SY)
350 IF S=0 OR S$<>" "THEN 380
360 DEF MOVE(7)=SPRITE(4,S,2,4,0,2):POSITION 7,PX*8+
    16,PY*8+24:MOVE 7
370 PLAY "O1#F0RO0#F:R"
380 GOSUB 400(企鹅与汽车相撞的判断及处理)
385 GOSUB 450(转移小企鹅的处理)
390 RETURN
400 REM "Collide with car"(企鹅碰汽车的处理)
410 IF CRASH(7)>=0 THEN PLAY "O1E0DCBAG:O0#A#G#
    FE#D#C":GOTO 60(转结束)
420 RETURN
450 REM "Move small penguin"(转移小企鹅)
460 IF PY=1 AND M=0 THEN H=H+1:M=1:SPRITE H:N=1:
    PLAY "O5C0DEDE:O3C0DEF#G"
470 IF PY=15 AND N=1 THEN M=0:SPRITE H,(H-1)*8+104,
    152:N=0:PLAY "O5#G0FEDC:O3#G0#FE#D#C"
480 IF H=5 AND N=0 THEN PLAY "O2G3GO3CCEECC:
    O1C3EGECGCC ": PLAY " GGAGEDC5O2C3O3C1:
    CEGECGCC1CC":GOTO 60
490 RETURN
```

程序说明：

(1)10 行中的 CGSET RND(2),0 的作用是为了使每次游戏时，随机改变背景的颜色。T 是时间限制变量，欲增加游戏难度，可改变 T 的赋值。

(2)25 行使程序每循环一次,时间值减 1,并打印出所剩的时间值。当 T=0 转结束处理。

(3)60 行的作用是使 8 个运动卡通(7 辆汽车和大企鹅)和 5 个显示卡通消失。

(4)70 行的作用是读 I# 操纵器的功能键,直到读到按 START、A、B、SELECT 中任一键后,或重新运行程序;或停机。

(5)110～135 行通过引入 P、Q 变量,仅用一次循环语句就画出了马路、人行道的南、北边界和中心线。

(6)150 行定义、显示 5 个小企鹅。

(7)170、180 行定义、显示、启动 7 辆汽车。

(8)260～360 行完成了用 1# 操纵器的方向键控制企鹅运动。K 是读到的方向键值,S 决定企鹅的运动方向,SX、SY 是限制企鹅在人行道内运动的变量,由它们保证企鹅运动不出界。

(9)460～480 行为转移小企鹅程序。M 为抱起小企鹅记忆变量,N 为放下小企鹅提示变量,通过 M、N 的控制保证了大企鹅一次只抱一个小企鹅和避免没有抱企鹅而在路南显示小企鹅。

程序 No. 7—4 中的模块 D、E、G 是通用模块,它们可以作为工具使用。

为便于读者对照,下面再列出《企鹅看外婆》游戏顺序结构的程序清单。

```
5 REM "No. 7—5 PENGUINS VISIT GRANDMOTHER"
10 CLS:CLEAR:SPRITE ON:T=800:CGSET RND(2),0
20 PLAY "V15T1:V15T1"
30 FOR I=0 TO 27:P=3:Q=15
40 IF I<10 OR I>16 THEN LOCATE I,9:PRINT CHR$(227)
50 IF I>=10 AND I<=16 THEN P=1:Q=17
60 LOCATE I,P:PRINT CHR$(197):LOCATE I,Q:PRINT CHR
   $(197):NEXT
70 FOR I=2 TO 16:LOCATE 10,I:PRINT CHR$(226):LOCATE
```

```
16,I:PRINT CHR$(226):NEXT
80 FOR I=1 TO 5:DEF SPRITE I,(RND(4)0,0,0,0)="defg":
   SPRITE I,(I-1)*8+104,20:NEXT
90 DEF MOVE(7)=SPRITE(4,0,1,2,0,2):POSITION 7,120,144:
   MOVE 7
100 FOR I=0 TO 6:DEF MOVE(I)=SPRITE(6,7,I+1,130,0,RND
   (4))
110 POSITION I,250,RND(60)+56:MOVE I:NEXT
120 T=T-1:LOCATE 22,21:PRINT T;"  "
130 IF T=0 THEN 340
140 FOR I=0 TO 6
150 IF MOVE(I)=0 THEN POSITION I,250,RND(60)+56:MOVE I
160 NEXT
170 S=0:SX=0:SY=0:K=STICK(0):IF K=0 THEN 280
180 IF K=1 THEN S=3:SX=1
190 IF K=2 THEN S=7:SX=-1
200 IF K=4 THEN S=5:SY=2
210 IF K=8 THEN S=1:SY=1
220 PX=(XPOS(7)-16)/8:IF PX=10 THEN PX=11
230 IF PX=15 THEN PX=14
240 PY=(YPOS(7)-24)/8
250 S$=SCR$(PX+SX,PY+SY)
260 IF S=0 OR S$<>" "THEN 280
270 DEF MOVE(7)=SPRITE(4,S,2,4,0,2):POSITION 7,PX*8+
   16,PY*8+24:MOVE 7
275 PLAY "O1#F0RO0#F:R"
280 IF CRASH(7)>=0 THEN PLAY "O1E0DCBAG:O0#A#G#
   FE#D#C"
290 IF PY=1 AND M=0 THEN H=H+1:M=1:SPRITE H:N=1:
   PLAY "O5C0DEDE:O3C0DEF#G"
300 IF PY=15 AND N=1 THEN M=0:SPRITE H,(H-1)*8+104,
```

```
    152:N=0:PLAY "O5#G0FEDC:O3#G0#FE#D#C"
310 IF H = 5 AND N = 0 THEN PLAY "02G3GO3CCEECC:
    O1C3EGECGCC " : PLAY " GGAGEDC5O2G3O3C1:
    CEGECGCC1CC":GOTO 340
320 GOTO 120
340 ERA 0,1,2,3,4,5,6,7:FOR I=1 TO 5:SPRITE I:NEXT:CLS
350 LOCATE 0,20:PRINT "Press START to continue"
360 S=STRIG(0):IF S=0 GOTO 350
370 IF S=1 THEN RUN
380 CLS:LOCATE 8,8:PRINT "GOOD—BYE!!!":LOCATE 0,20:
    END
```

对比程序 No. 7—4、No. 7—5，显而易见，结构化程序层次分明，便于阅读理解，但程序量较大；顺序结构，顺序编写、程序紧凑、阅读理解要困难些。

另外，在程序 No. 7—4 中，模块 B 只被主程序调用一次，完全可以把它放入管理程序中。这样调整后，No. 7—4 就变为混合结构的程序，而且，调整后并不影响整个程序的层次和易读性。所以说混合结构是较优化的程序结构。

7.4 游戏程序的设计过程

我们知道，机械行业在开发新产品时，首先要进行市场需求调查，根据市场调查进行产品性能需求分析，确定产品的功能要求；然后根据功能要求，进行图纸、资料和各种技术细节的设计；在图纸、资料完善的基础上再进行制造和装配；装配完成后还需进行调试、运行实验，只有产品完全合格后才能出厂。由此可知，在机械工业中最终的产品不仅仅是机器产品本身，它还包括一整套的技术资料(图纸、技术说明书、使用说明书、维护手册等)。归纳起来，一种新产品的开发共有需求分析、设计、制造、调试和运行五个工程

阶段。

对于计算机的软件开发过程，电脑专家们也提出了工程化的管理方案。即软件的开发也要经过需求分析、设计、编程、测试和运行五个阶段。同机械产品一样，一项软件产品的技术资料也要包含系统说明书、模块说明书、程序清单、测试文件和使用维护手册五个组成部分。

因为游戏软件是一种特殊的应用软件，所以游戏程序的设计过程同样要经过以上五个阶段。也要具备一般软件的技术资料。由此可见，前三节仅讨论了游戏程序设计过程的第二、三两个阶段。一般情况下，多数读者设计游戏程序仅是兴趣、爱好所致，不是搞盈利性的软件开发，市场调查、分析可不予考虑。但是，分析阶段形成的书面资料——系统说明书还是必须的，因为它是设计、编程、测试阶段的依据。为系统起见，我们仍以软件开发的五个阶段为内容，以《成龙救金凤》游戏为例介绍游戏程序的设计过程。

7.4.1 分析阶段

分析阶段首先必须集中精力、认真地分析开发该软件系统的目的。要选择恰当的游戏类型；要确定游戏的具体功能目标；要考虑游戏的玩法，游戏的参加者的数量，游戏者可有几个对手，电脑是否充当对手；还要决定是否奖励优秀和处罚不好的游戏者等等。然后确定游戏的功能目标。对于《成龙救金凤》游戏的功能目标如下：

1. 游戏开始，屏幕出现金凤被秃鹰叨走的画面。

2. 随后显示游戏主画面。屏幕上画出七层楼房，九个藏宝箱散落在七层楼中，成龙位于楼底层左边入口处，金凤位于楼上右上角，妖怪显示于 3～7 层楼之间。

3. 游戏者使用 1# 操纵器方向键控制成龙运动，并伴有脚步声。

4. 成龙必须按电脑约定的顺序依次打开九个箱子。每打开一

个箱子记录10分。当积分达到90分时，游戏者胜。显示成龙救出金凤回家的画面。并伴有祝贺音乐。

5. 当游戏者被妖怪抓住或者时间到了成龙没有全部打开箱子，则发声、游戏结束，转结束或继续游戏选项画面。

上述游戏程序的功能目标，就是一份简单的系统说明书。

7.4.2 设计阶段

设计阶段要求设计出符合系统说明书要求的软件系统方案，并从几个方案中优选出较好的方案。方案确定后，将系统划分成几个具有确定功能的模块，并考虑处理好模块间传送的数据与类型、模块间的调用关系、选择出较好的模块结构图、写出各模块的功能说明书。另外，在模块结构图中还必须对所有模块编号，简单说明各模块的功能，必要时还要标出每个模块的输入、输出内容。

《成龙救金凤》游戏的模块结构图见图7—17。图中A为管理程序；B为游戏开始时的调用程序，以后各场不再调用；C、D、E为每场游戏开始时的调用模块；F为救人成功后的调用模块；(B、C、D、E、F均为条件调用模块，)G、H、I为循环调用模块，这三个模块构成游戏的主过程；K为结束处理模块，仅在时间到了没有打开箱子、成龙被妖怪抓住和救人成功三种情况下才调用。由于各模块功能比较明确，已在模块结构图中标出，故不再分模块作功能说明，留给读者练习。

7.4.3 编程阶段

编程阶段的任务就是为每个模块编写程序，将模块说明书转换成用某种程序设计语言书写的源程序。为使读者能够编写出逻辑上正确、结构合理、层次分明、思路清晰、便于阅读理解的程序，我们特介绍由顶向下，逐步求精的程序设计方法。其基本原则是：以模块说明书为基础，对每个模块的抽象算法由粗至精(即由抽象到具体)，逐步精确化，逐步扩充细节，直到成为一个能在机器上执行的程序。

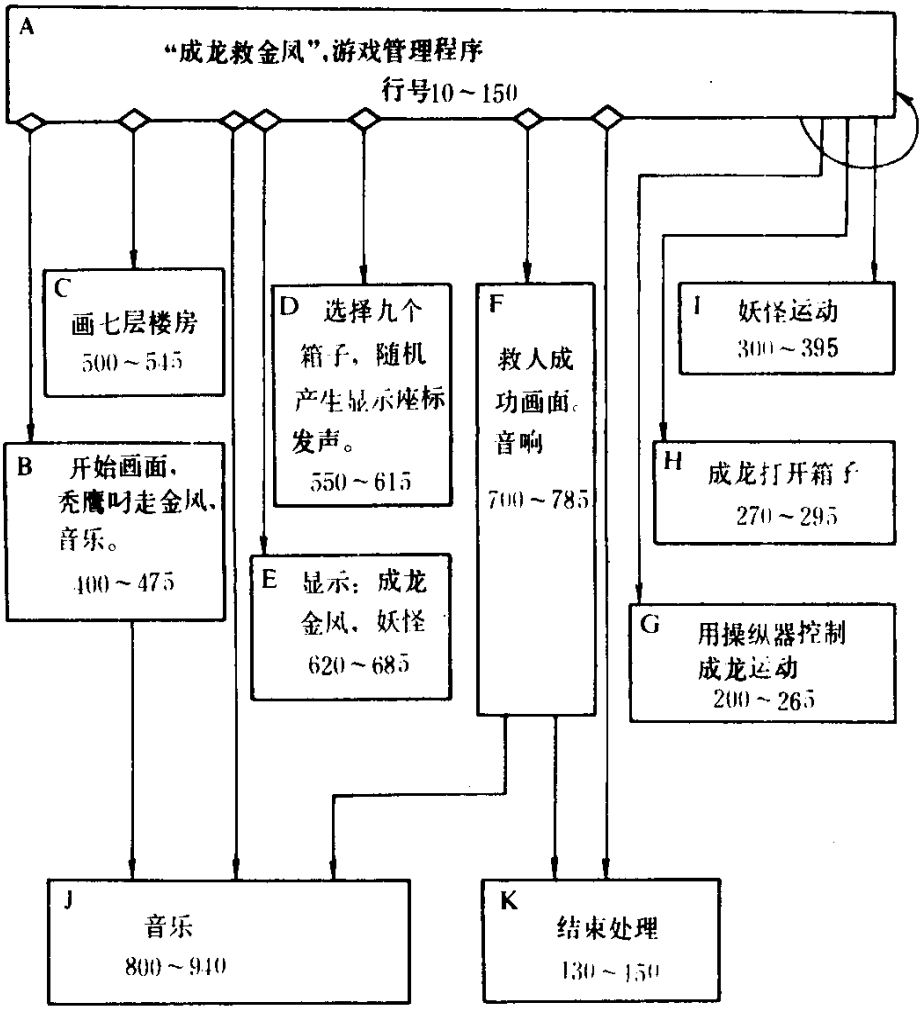


图 7—17 《成龙救金凤》游戏程序模块结构图

1. 模块的逐步求精

逐步求精要遵循以下四点：

(1)以合理的程序结构为目标(程序结构清晰、易读、易理解、易修改)。

(2)严格由粗到精的顺序，关于细节的决定尽可能地向后推迟，以避免整体结构与局部细节的纠缠，使细节的修改不影响上一级的抽象结构，因而程序的易修改性好。

(3)程序细节精细化的同时进行数据结构的精细化。

(4)一边编写程序，一边检查其正确性，以提高程序的可靠性，减少返工。

最后还要强调一点，逐步求精的过程要直到所有模块都已足

够简单、能够使用规定的语言编写程序时才可终止。

2. 用子程序做公共模块

在程序的编写过程中，要注意把许多模块都调用的公用模块作为子程序来编写，以便于调用。对公用模块，必须高质量地编写好程序。这样，既可使主程序的逻辑结构清晰，又可使程序的可靠性和效率提高。

3.《成龙救金凤》游戏程序的详细设计

根据各模快的功能分配，分别画出流程图，做好行号分配。再根据流程图，编写出实现功能要求的源程序。

(1)模块 B：

画出开始画面。时间：傍晚。太阳刚刚落山，残霞满天，暗红色的背景。地点：郊外。画出一条路，路的左端有一棵树，天上有一朵白云。在悠扬的音乐声中，成龙与金凤手拉手在郊外散步。当走到白云下方时，从云中冲出埋伏的秃鹰，叼走金凤，天空变暗，成龙尾随狂追，音乐紧促、激烈。流程图如图 7—18。

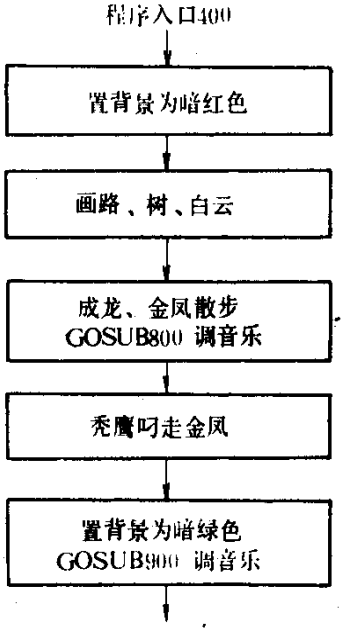


图 7—18　模块 B 流程图

对照图 7—18，写出详细程序如下：(行号分配：从 400 行开

始)

```
400 REM "BEGIN PICTURE"
405 CGSET 0,1:SPRITE ON:CLS:PALETB 0,8,48,47,47
410 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(197):NEXT
415 LOCATE 10,2:PRINT CHR$(216);CHR$(218):LOCATE 8,
    3:PRINT CHR$(216);CHR$(217);CHR$(218)
420 LOCATE 1,8:PRINT CHR$(208):LOCATE 0,9:PRINT CHR
    $(249);CHR$(203);CHR$(251):LOCATE 1,10:PRINT CHR
    $(238):LOCATE 1,11:PRINT CHR$(212)
425 LOCATE 2,16:PRINT "CHENG LONG rescues JIN FENG":LO-
    CATE 8,20:PRINT "BY        YU CHUN"
430 DEF MOVE(4)=SPRITE(0,7,8,90,0,3):POSITION 4,210,
    100:MOVE 4
435 DEF MOVE(5)=SPRITE(1,7,8,80,0,3):POSITION 5,220,
    104:MOVE 5
440 PLAY "M0V12M1V12Y1T2:M0V9T2:M1V2T2":FF=1:GOSUB
    800
445 DEF MOVE(6)=SPRITE(15,5,1,23,1,0):POSITION 6,88,48:
    MOVE 6
450 IF CRASH(6)<>5 THEN 450
455 DEF MOVE(5)=SPRITE(1,2,1,58,0,3):POSITION 5,70,104:
    MOVE 5
460 DEF MOVE(6)=SPRITE(15,2,1,55,0,0):POSITION 6,76,96:
    MOVE 6
465 PALETB 0,10,48,47,47
470 DEF MOVE(4)=SPRITE(0,3,1,94,0,3):POSITION 4,60,100:
    MOVE 4
475 GOSUB 900:ERA 4
480 RETURN
```

(2)模块 C

画七层楼房。用背景图形库中E4号画地板，用H3号画楼梯。使用变量X、Y、B$、J$。流程图见图7—19。

对照流程图，编程如下：

分配起始行号500

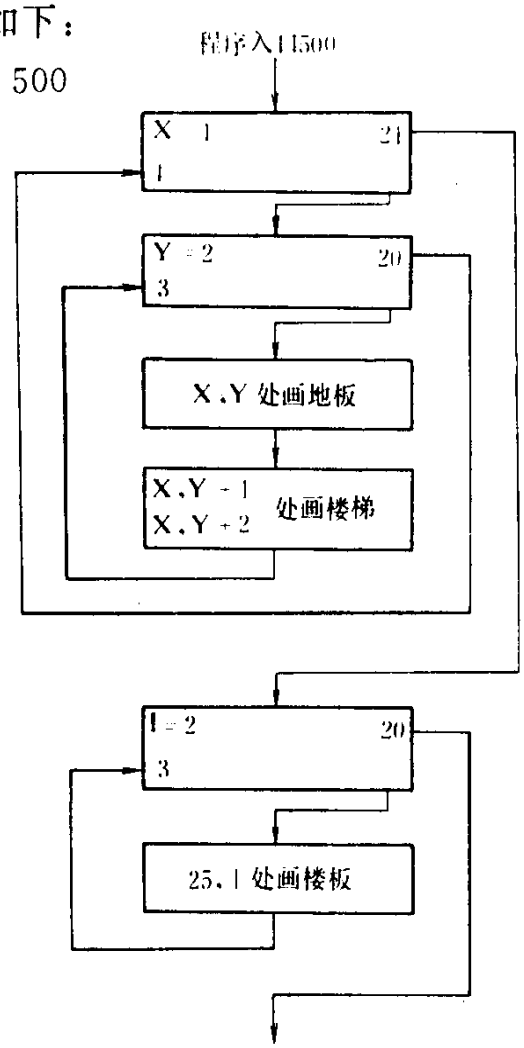


图7—19　模块C流程图

```
500 REM "BUILDING"
505 B$=CHR$(187):J$=CHR$(210):J$=J$+"          "+J$
510 FOR X=1 TO 21 STEP 4
515 FOR Y=2 TO 20 STEP 3
520 LOCATE X,Y:PRINT B$;B$;B$;B$;IF Y>19 THEN 530
525 LOCATE X,Y+1:PRINT J$:LOCATE X,Y+2:PRINT J$
530 NEXT:NEXT
535 FOR I=2 TO 20 STEP 3
```

```
540 LOCATE 25,I:PRINT B$:NEXT
545 RETURN
```

(3)模块 D

选用 E7～F6,H5～I4,L5～M4 共 27 个背景图案为随机选用的藏宝箱。把每次选定的 9 个图形按顺序存入 PS$ 串变量中,然后随机产生在楼板上的 9 个藏宝箱的座标,显示 9 个背景图形。这 9 个图形只能显示在楼板上,不能显示在楼梯上或其它空间。

使用变量:PS$、P1、P2、PL、PS、S$、SX、SY

其中:PS$:用于储存 9 个藏宝箱的图形。

P1、P2:选定藏宝箱的图形。

PL:PS$ 的字符长度。

PS:藏宝箱的顺序号。

S$:藏宝箱位置的确定辅助变量,由它保证各藏宝箱只显示在楼板上。

SX、SY:藏宝箱的打印座标。

流程图见图 7—20

对照流程图编写程序如下:

程序入口:起始行号 550

```
550 REM "BOXES"
555 P1=RND(3)+5
560 C=80:IF P1=7 THEN C=90
565 P2=P1*22+C:PS$=CHR$(P2)
570 FOR I=1 TO 8
575 PS$=PS$+CHR$(P2+I):NEXT
580 PL=LEN(PS$):PS=1
585 FOR S=1 TO PL
590 SX=2+RND(24):SY=2+RND(18)
595 S$=SCR$(SX,SY+1)
600 IF(S$=" ")OR(INSTR(PS$,S$)>0)OR((SX-1)MOD 4=0)
```

```
    THEN 590
605 LOCATE SX,SY:PRINT MID$(PS$,S,1)
610 NEXT
615 RETURN
```

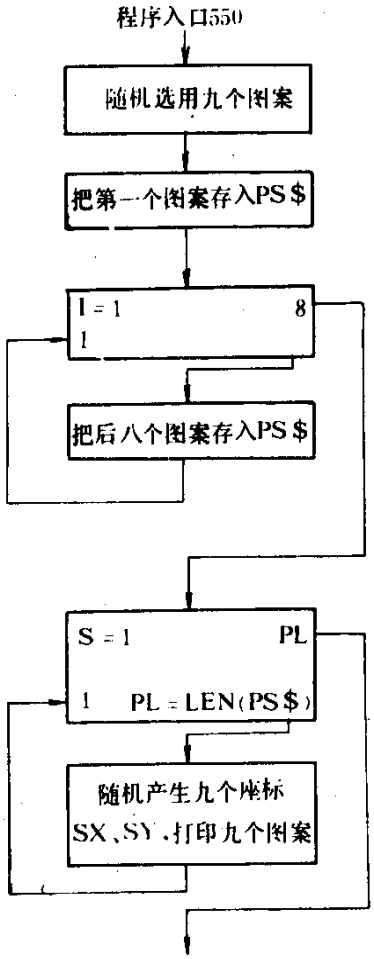


图 7—20　模块 D 流程图

(4)模块 E

在(200,16)处显示金凤，在(12,176)处显示成龙，在(92,32)～(220,104)之间显示妖怪。使用 E(KN－1)数组定义妖怪的数量，欲增加游戏难度可调整 KN 的数值，KN 最大取值为 6。

该模块较简单，故略去流程图，直接写出程序。

程序入口为 620

```
620 REM "SHOW SPRITE"
625 LOCATE 12,0:PRITN "                    "
630 DEF SPRITE 3,(2,1,0,0,0)="$%&'":SPRITE 3,200,16
635 T=0:F=0:KN=2:KX=0:KY=0:KD=0:TI=800
640 LOCATE 0,22:PRINT "      SCORE ";T
645 DEF MOVE(7)=SPRITE(0,0,2,4,0,RND(4)):POSITION 7,
    12,168:MOVE 7
650 DIM E(KN-1)
655 FOR F=0 TO KN-1
660 DEF MOVE(F)=SPRITE(11,1,2,4,0,RND(4))
665 SX=10+RND(17):SY=1+RND(10)
670 IF SCR$(SX,SY)=" "OR SCR$(SX,SY-1)=" "GOTO 665
675 POSITION F,SX*8+12,SY*8+16:MOVE F
680 NEXT
685 RETURN
```

(5)模块 G

用 1# 操纵器控制成龙上、下楼和左、右移动。与例 7.3 中的模块 D 大同小异，可直接写出程序。

程序入口行号为 200。

```
200 REM "CONTROL"
205 S=0:SX=0:SY=0
210 K=STICK(0):IF K=0 GOTO 265
215 IF K=1 THEN S=3:SX=1
220 IF K=2 THEN S=7:SX=-1
225 IF K=4 THEN S=5:SY=1
230 IF K=8 THEN S=1:SY=-1
235 PX=(XPOS(7)-12)/8:PY=(YPOS(7)-16)/8
240 S$=SCR$(PX+SX,PY+SY+1):S1$=SCR$(PX+SX,PY
    +SY)
245 IF S=0 OR S$=" "GOTO 265
```

```
250 DEF MOVE(7)=SPRITE(0,S,1,4,0,RND(4)):POSITION 7,
    PX*8-12,PY*8+16:MOVE 7
255 IF S=3 OR S=7 THEN PLAY "O0=F0RO1=F":GOTO 265
260 PLAY "O1=F0RO0=FR"
265 RETURN
```

(6)模块 H

成龙若按 PS$ 中记录的顺序打开箱子，则发声，打印积分，并令箱子消失。每打开一个箱子加 10 分。使用 PX、PY 表示成龙在背景画面中的座标。若 9 个箱子全部打开则转救人成功程序。流程图见图 7—21。

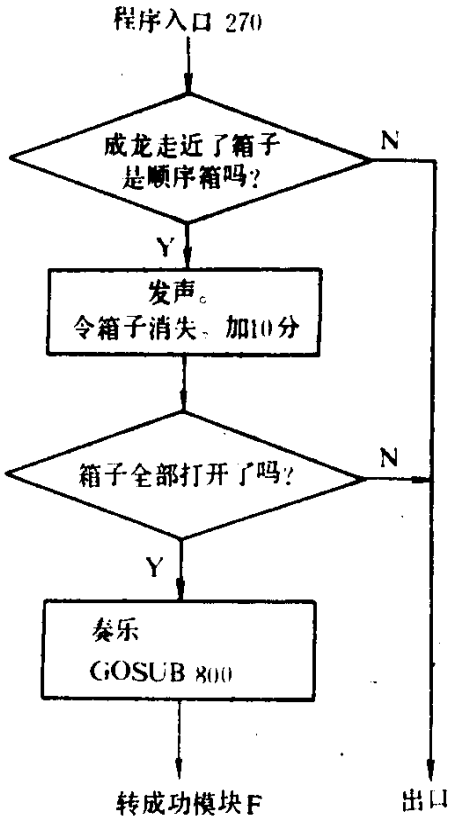


图 7—21　模块 H 流程图

程序入口起始行号为 270。

```
270 REM "OPEN BOX"
275 IF S1$<>MID$(PS$,PS,1)THEN 295
280 PLAY "OAC0=CD=DEF=FG=G"
```

```
285 LOCATE PX+SX,PY+SY:PRINT " ":T=T+10:LOCATE 8,
    22:PRINT T
290 PS=PS+1:IF PS>PL THEN PLAY "O4C3DEFG:O4E3FGAB":
    GOTO 110(成功处理)
295 RETURN
```

(7)模块 I

妖怪自动运动模块。轮流测试某个妖怪在背景画面中的座标。测试妖怪前后是否是楼板，上、下是否是楼梯。使妖怪在楼板上左、右移动，在楼梯上上、下移动，移动方向为 KD。流程图见图 7—22。

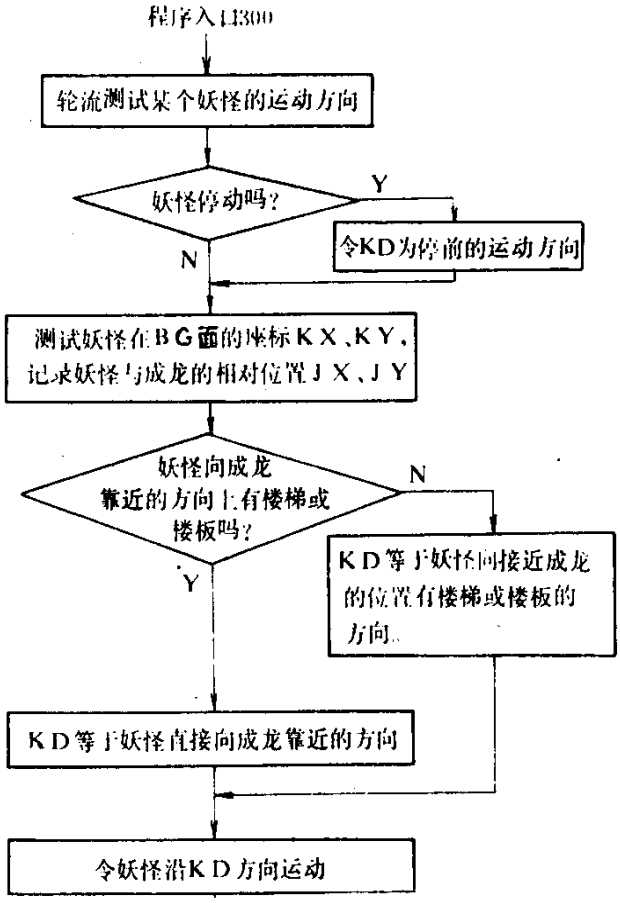


图 7—22　模块 I 流程图

对照流程图编写程序如下：

程序入口行号为 300

```
300 REM "BOGY MOVEMENT"
305 F=F+1:IF F>KN-1 THEN F=0
310 KD=VCT(F):IF KD>0 GOTO 320
315 KD=E(F)
320 KX=(XPOS(F)-12)/8:JX=SGN(PX-KX)
325 KY=(YPOS(F)-16)/8:JY=SGN(PY-KY+1)
330 SK$=SCR$(KX,KY):SR$=SCR$(KX+1,KY):SL$=SCR
    $(KX-1,KY):SD$=SCR$(KX,KY+1):SU$=SCR$(KX,
    KY-1)
335 IF (SR$+SL$<>" ") AND (SD$+SU$<>" ") GOTO 345
340 IF (KD=3 AND JX=1) OR (KD=7 AND JX=-1) OR (KD=5
    AND JY=1) OR (KD=1 AND JY=-1)GOTO 390
343 KD=KD+4+8*(KD>3):GOTO 390
345 IF JX*JY<>0 AND RND(2)=1 GOTO 355
350 IF JX=1 AND SR$<>" "THEN KD=3:GOTO 390
353 IF JX=-1 AND SL$<>" "THEN KD=7:GOTO 390
355 IF JY=1 AND SD$<>" "THEN KD=5:GOTO 390
360 IF JY=-1 AND SU$<>" "THEN KD=1:GOTO 390
365 S=RND(4)+1:ON S GOTO 370,375,380,385
370 IF SR$<>" "THEN KD=3:GOTO 390
375 IF SL$<>" "THEN KD=7:GOTO 390
380 IF SD$<>" "THEN KD=5:GOTO 390
385 IF SU$<>" "THEN KD=1
390 DEF MOVE(F)=SPRITE(11,KD,1,4,0,RND(4)):POSITION
    F,KX*8+12,KY*8+16:MOVE F
393 E(F)=KD
395 RETURN
```

(8)模块 F

救人成功处理模块。令所有定义的卡通消失。画室内地板。定义成龙、金凤相向奔跑、拥抱。成龙起始座标(0,104),金凤起始座标(240,104)。音乐停止后,清屏,画路,成龙和金凤手拉手向屏幕左方行走。成龙起点座标为(220,104),金凤的起点座标为(230,104)。

流程图见图 7—23。

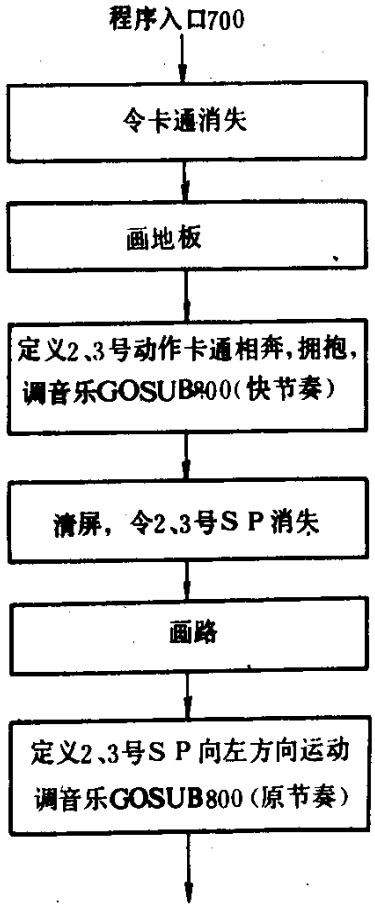


图 7—23　模块 F 的流程图

对照流程图编写程序如下:

程序入口行号为 700

```
700 REM "VICTORY1"
705 ERA 0,1,2,3,4,5,6,7:SPRITE 3:CLS
```

```
710 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(208);:NEXT
715 PALETB 0,10,10,21,49
720 FOR I=2 TO 3
725 DEF MOVE(I)=SPRITE(I-2,4*I-5,2,58,0):POSITION I,(I
    -2)*240,104:MOVE I:NEXT
730 PLAY "M0V12M1V9Y2T1:M0V12T1:M1V0T1":FF=0:GOSUB
    800
750 REM "VICTORY2"
755 ERA 2,3:CLS:PALETB 0,5,22,39,56
760 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(197);:NEXT
765 FOR I=2 TO 3
770 DEF MOVE(I)=SPRITE(I-2,7,4,230,0)
775 POSITION I,(I-3)*8+230,104:MOVE I:NEXT
780 PLAY "M0V12M1V12Y1T4:M0V9T4:M1V2T4":FF=1:GOSUB
    800
785 RETURN
```

(9)模块 K

结束处理。若是成龙被妖怪抓住或者时间到了则打印游戏结束，询问是否继续游戏。若救人成功，在显示成功画面后，直接询问是否继续。两种情况用一个选项程序，即读 I²操纵器的“START”键和“SELECT”键。若继续游戏则按“START”；否则按“SELECT”。

模块 K 的流程图和程序清单见管理程序。

(10)模块 J

音乐模块。为简化程序，成龙与金凤散步的音乐、每场的开始音乐和救人成功后的音乐用同一段乐章。不同的是在成龙与金凤相向奔跑时要求的激动热烈的音乐是通过把音乐节奏加速一倍而实现的。其它则使用正常的节奏。该段音乐为主题音乐，称为“MUSIC1”。

在飞鹰叨走金凤时有一段急促紧迫的音乐，称为"MUSIC2"。两段音乐的程序如下：

MUSIC 1

```
800 REM "MUSIC 1"
805 PLAY "O4G5E3GA4G1:O4C5O3GA:O1C3CC5C3C"
810 PLAY "E3DCDE4D1:GDG:C5C3CC5"
815 PLAY "E3G7G3E4G1:AGGO4C:C3CC5C3CC5"
820 PLAY "A3O5C6O4G3A:DCO3G:C3CC5C3C"
825 PLAY "DE1DC7:CEC:C5C3CC5"
830 PLAY "G5E3GA4G1:O4CO3GA:G3GG5G3G"
835 PLAY "E3DCDE4D1:GDG:G5G3GG5"
840 PLAY "E3G7G3E4G1:AGGO4C:G3GG5C3CC5"
845 PLAY "A3O5C6D3G:ECD:C3CC5C3C"
850 PLAY "DE1DC7:GDC:C5C3CC5"
855 IF FF=0 THEN FF=1:GOTO 805
860 RETURN
```

MUSIC 2

```
900 REM "MUSIC 2"
905 PLAY "M0V10M1V8Y3T2:M0V3T2:T2"
910 PLAY "O4C3CO3GA:O3C3CO2GA:O2C5G"
915 PLAY "O4CCO3GADG6:O3CCO2GADG6:CGDG"
920 PLAY "D3G6A3O4E6:D3G6A3O3E6:DGAO3E"
925 PLAY "O3A3O4E6:O2A3O3E6:O2AO3E"
930 PLAY "C3CO3GA:C3CO2GA:O2CG"
935 PLAY "O4CCO3GA:O3CCO2GA:CG"
940 RETURN
```

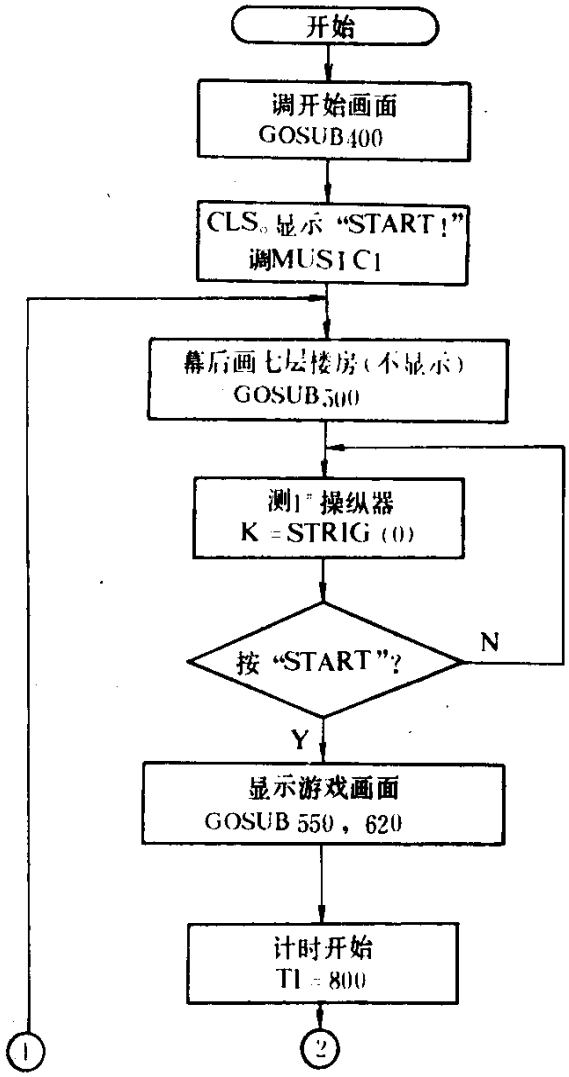
(11)模块 A

管理程序。由它完成整个游戏过程的控制。流程图见图 7—24。

对照程序框图编写程序如下：

程序入口行号 10

```
10 REM "MAIN PROGRAM"
15 GOSUB 400 (开始画面)
20 CLEAR:CLS:CGSET RND(2),RND(3):CGEN 2
25 LOCATE 8,8:PRINT "START!!!"
30 SCREEN 0,1:PLAY "M0V12M1V12Y1T3;M0V9T3;M1V2T3":
   FF=1:GOSUB 800
35 SPRITE ON:PLAY "V15T1:V15T1"
```



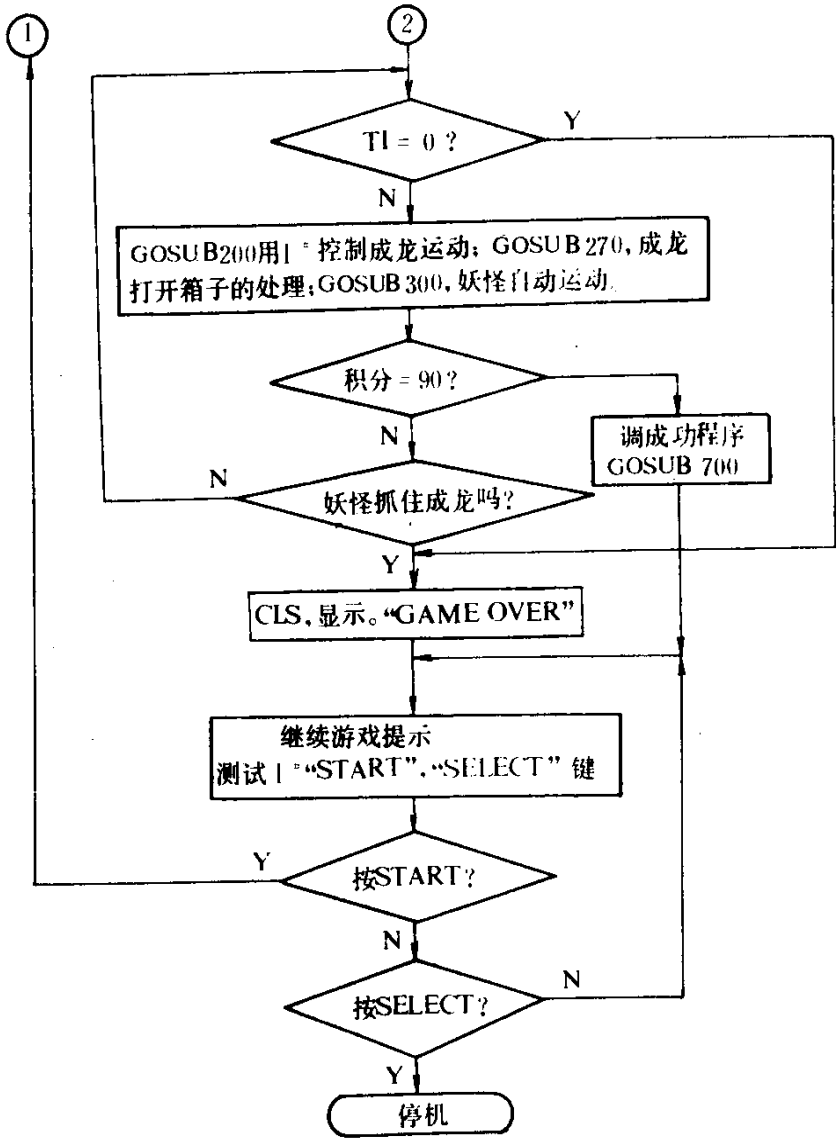


图 7—24　模块 A(管理程序)流程图

```
40 IF SCR$(1,2)<>" "GOTO 50
45 GOSUB 500 (画七层楼)
50 SCREEN 0,0:VIEW:PRINT "      TIME =                PRESS
   START"
55 K=STRIG(0):IF K=0 GOTO 55
60 PLAY "O3C1EE:O3E1FG"
65 GOSUB 550 (显示九个箱子)
```

```
70 GOSUB 620 (显示成龙、妖怪、金凤)
75 TI=TI-1:LOCATE 7,0:PRINT TI"      "
80 IF TI=0 THEN 115
85 GOSUB 200 (用操纵器控制成龙运动)
90 GOSUB 270 (成龙打开箱子的处理)
95 GOSUB 300 (妖怪自动运动)
100 IF CRASH(7)>=0 THEN PLAY "O1B2AGFEDC":GOTO 115
105 GOTO 75
110 GOSUB 700:GOTO 130 (救人成功画面)
115 CLS:LOCATE 9,6:PRINT "GAME OVER!"
120 SPRITE OFF
130 LOCATE 2,20:PRINT"CONTINUE/start END/select"
135 S=STRIG(0):IF S=0 THEN 135
140 IF S=1 THEN RUN 20
145 IF S=2 THEN CLS:LOCATE 10,4:PRINT "GOOD BYE!!!":
    LOCATE 0,21:END
150 GOTO 135
```

现在《成龙救金凤》游戏的全部程序编写完毕，只要把各模块的程序合成即得模块结构的程序清单(见附录)。

为便于对照，下面再给出顺序结构的《成龙救金凤》程序清单。由于编程思路一样，不多作说明，请读者自己分析。

例 7.4 《成龙救金凤》顺序结构程序清单

```
5 REM "No. 7—7"
10 REM "CHENG LONG rescues JIN FENG"
15 CGSET 0,1:SPRITE ON:CLS:PALETB 0,8,48,47,47
20 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(197):NEXT
25 LOCATE 10,2:PRINT CHR$(216);CHR$(218):LOCATE 8,3:
   PRINT CHR$(216);CHR$(217);CHR$(218)
30 LOCATE 1,8:PRINT CHR$(208):LOCATE 0,9:PRINT CHR
   $(249);CHR$(203);CHR$(251):LOCATE 1,10:PRINT CHR
```

```
$(238):LOCATE 1,11:PRINT CHR$(212)
35 LOCATE 2,16:PRINT "CHENG LONG rescues JIN FENG":LO-
   CATE 8,20:PRINT "BY        YU      CHUN"
40 DEF MOVE(4)=SPRITE(0,7,8,90,0,3):POSITION 4,210,100:
   MOVE 4
45 DEF MOVE(5)=SPRITE(1,7,8,80,0,3):POSITION 5,220,104:
   MOVE 5
50 PLAY "M0V12M1V12Y1T2:M0V9T2:M1V2T2":FF=1:GOSUB
   800
55 DEF MOVE(6)=SPRITE(15,5,1,23,1,0):POSITION 6,88,48:
   MOVE 6
60 IF CRASH(6)<>5 THEN 60
65 DEF MOVE(5)=SPRITE(1,2,1,58,0,3):POSITION 5,70,104:
   MOVE 5
70 DEF MOVE(6)=SPRITE(15,2,1,55,0,0):POSITION 6,76,96:
   MOVE 6
75 PALETB 0,10,48,47,47
80 DEF MOVE(4)=SPRITE(0,3,1,94,0,3):POSITION 4,60,100:
   MOVE 4
85 PLAY "M0V10M1V8Y3T2:M0V3T2:T2"
90 PLAY "O4C3CO3GA:O3C3CO2GA:O2C5G"
95 PLAY "O4CCO3GADG6:O3CCO2GADG6:CGDG"
100 PLAY "D3G6A3O4E6:D3G6A3O3E6:DGAO3E"
105 PLAY "O3A3O4E6:O2A3O3E6:O2AO3E"
110 PLAY "C3CO3GA:C3CO2GA:O2CG"
115 PLAY "O4CCO3GA:O3CCO2GA:CG"
120 ERA4
125 CLEAR:CLS:CGSET RND(2),RND(3):LOCATE 8,8:PRINT
    "START!!!"
130 SCREEN 0,1:PLAY "M0V12M1V12Y1T3:M0V9T3:M1V2T3":
    FF=1:GOSUB 800
```

```
135 SPRITE ON:PLAY "V15T1:V15T1"
140 IF SCR$(1,2)<>" "GOTO 170
145 B$=CHR$(187):J$=CHR$(210):J$=J$+"        "+J$
150 FOR X=1 TO 21 STEP 4:FOR Y=2 TO 20 STEP 3:LOCATE
    X,Y:PRINT B$;B$;B$;B$:IF Y>19 THEN 160
155 LOCATE X,Y+1:PRINT J$:LOCATE X,Y+2:PRINT J$
160 NEXT:NEXT
165 FOR I=2 TO 20 STEP 3:LOCATE 25,I:PRINT B$:NEXT
170 SCREEN 0,0:VIEW:PRINT "      TIME=              PRESS
    START".
175 K=STRIG(0):IF K=0 GOTO 175
180 PLAY "O3C1EE:O3E1FG"
185 P1=RND(3)+5:C=80:IF P1=7 THEN C=90
190 P2=P1*22+C:PS$=CHR$(P2)
195 FOR I=1 TO 8:PS$=PS$+CHR$(P2+I):NEXT
200 PL=9:PS=1
205 FOR S=1 TO PL
210 SX=2+RND(24):SY=2+RND(18):S$=SCR$(SX,SY+1)
215 IF S$=" "OR INSTR(PS$,S$)>0 OR (SX-1)MOD 4=0
    TNEN 210
220 LOCATE SX,SY:PRINT MID$(PS$,S,1):NEXT
225 LOCATE 12,0:PRINT "                    "
230 DEF SPRITE 3,(2,1,0,0,0)="$%&'":SPRITE 3,200,16
235 T=0:F=0:KN=2:KX=0:KY=0:KD=0:TI=800:LOCATE 0,
    22:PRINT "      SCOR ";T
240 DEF MOVE(7)=SPRITE(0,0,2,4,0,RND(4)):POSITION 7,
    12,168:MOVE 7
245 DIM E(KN-1)
250 FOR F=0 TO KN-1:DEF MOVE(F)=SPRITE(11,1,2,4,0,
    RND(4))
255 SX=10+RND(17):SY=1+RND(10):IF SCR$(SX,SY)=" "
```

```
    OR SCR$(SX,SY-1)=" " GOTO 255
260 POSITION F,SX*8+12,SY*8+16:MOVE F:NEXT
265 TI=TI-1:LOCATE 7,0:PRINT TI"          ":IF TI=0 THEN
    495
270 S=0:SX=0:SY=0
275 K=STICK(0):IF K=0 THEN 320
280 IF K=1 THEN S=3:SX=1
285 IF K=2 THEN S=7:SX=-1
290 IF K=4 THEN S=5:SY=1
295 IF K=8 THEN S=1:SY=-1
300 PX=(XPOS(7)-12)/8:PY=(YPOS(7)-16)/8:S$=SCR
    $(PX+SX,PY+SY+1):S1$=SCR$(PX+SX,PY+SY):IF S
    =0 OR S$=" " GOTO 320
305 DEF MOVE(7)=SPRITE(0,S,1,4,0,RND(4)):POSITION 7,
    PX*8+12,PY*8+16:MOVE 7
310 IF S=3 OR S=7 THEN PLAY "O0#F0RO1#F": GOTO 320
315 PLAY "O1#F0RO0#FR"
320 IF S1$<>MID$(PS$,PS,1) THEN 335
325 PLAY "O4C0#CD#DEF#FG#G":LOCATE PX+SX,PY+
    SY:PRINT" ":T=T+10:LOCATE 8,22:PRINT T
330 PS=PS+1:IF PS>PL THEN PLAY"O4C3DEFG:O4E3FGAB":
    GOTO 445
335 F=F+1:IF F>KN-1 THEN F=0
340 KD=VCT(F):IF KD>0 GOTO 350
345 KD=E(F)
350 KX=(XPOS(F)-12)/8:JX=SGN(PX-KX):KY=(YPOS(F)-
    16)/8:JY=SGN(PY-KY+1)
355 SK$=SCR$(KX,KY):SR$=SCR$(KX+1,KY):SL$=SCR
    $(KX-1,KY):SD$=SCR$(KX,KY+1):SU$=SCR$(KX,
    KY-1)
360 IF SR$+SL$<>" " AND SD$+SU$<>" "GOTO 380
```

```
365 IF(KD=3 AND JX=1) OR (KD=7 AND JX=-1) OR (KD)=5
    AND JY=1) OR (KD=1 AND JY=-1) GOTO 430
370 KD=KD+4+8*(KD>3):GOTO 430
380 IF JX*JY<>0 AND RND(2)=1 GOTO 395
385 IF JX=1 AND SR$<>" " THEN KD=3:GOTO 430
390 IF JX=-1 AND SL$<>" " THEN KD=7:GOTO 430
395 IF JY=1 AND SD$<>" " THEN KD=5:GOTO 430
400 IF JY=-1 AND SU$<>" " THEN KD=1:GOTO 430
405 S=RND(4)+1:ON S GOTO 410,415,420,425
410 IF SR$<>" " THEN KD=3:GOTO 430
415 IF SL$<>" " THEN KD=7:GOTO 430
420 IF SD$<>" " THEN KD=5:GOTO 430
425 IF SU$<>" " THEN KD=1
430 DEF MOVE(F)=SPRITE(11,KD,4,0,RND(0)):POSITION F,
    KX*8+12,KY*8+16:MOVE F:E(F)=KD
435 IF CRASH(7)>=0 THEN PLAY "O1B2AGFEDC":GOTO 495
440 GOTO 265
445 REM"save VICTORY"
450 ERA 0,1,2,3,4,5,6,7:SPRITE 3:CLS
455 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(208):NEXT
460 PALET B 0,10,10,21,49
465 FOR I=2 TO3:DEF MOVE(I)=SPRITE(I-2,4*I-5,2,58,0):
    POSITION I,(I-2)*240,104:MOVE I:NEXT
470 PLAY "M0V12M1V9Y2T1:M0V12T1:M1V0T1":FF=0:GOSUB
    800
475 ERA 2,3:CLS:PALETB 0,5,22,39,56
480 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(197):NEXT
485 FOR I=2 TO 3:DEF MOVE(I)=SPRITE(I-2,7,4,230,0):PO-
    SITION I,(I-3)*8+230,104:MOVE I:NEXT
490 PLAY "M0V12M1V12Y1T4:MOV9T4:M1V2T4":FF=1:GO-
    SUB 800:GOTO 500
```

```
495 CLS:LOCATE 9,6:PRINT "GAME OVER":SPRITE OFF
500 LOCATE 2,20:PRINT "CONTINUE/start END/select"
505 S=STRIG(0):IF S=0 THEN 505
510 IF S=1 THEN RUN 125
515 IF S=2 THEN CLS:LOCATE 10,4:PRINT "GOOD BYE!!!":
    LOCATE 0,21:END
520 GOTO 505
800 (MUSIC 1)同结构程序中的 800~860(从略)。
```

7.4.4 测试阶段

测试也称为调试，它是通过人工分析程序或者使用测试数据执行已编好的程序来发现程序中存在错误的一种有效手段。一般来说，程序编写完毕，仅仅完成程序设计工作的一半，较大的工作量将在测试阶段。

测试工作一般分两步进行。首先进行分模块测试，待各个模块均测试通过后再进行整个程序的测试。

1. 分模块测试

分模块测试就是要对每一个模块的输入、输出进行测试。通过测试把各模块中隐含的各种错误找出来，予以纠正。也就是说，分模块测试就是首先集中测试程序的较小的组成部分，而不是一开始就把整个程序作为一个整体进行测试。这样做的好处是：第一、使排错工作(即找出并改正各模块中的错误的过程)变得简单易行；第二、使得由若干个模块组成的模块组的测试过程变得较为容易；第三、可以由几个人同时测试不同的模块，从而加快了测试工作的进程。

2. 整个程序的测试

测试整个程序的方法有两种：增量测试法和非增量测试法。

所谓增量测试就是把一个测试好的模块同另一个测试好的模块结合后进行测试，测试通过后再联一个模块，再测试。直到所有模块联合完毕，测试通过，则整个程序测试完成。

非增量测试则是把所有测试好的模块联接以后再统一测试。这种测试方法排错较困难，往往一个模块出现错误要殃及整个程序不能运行。所以，一般不用这种方法。

下面介绍用增量测试法，测试《成龙救金凤》游戏程序的工作过程(以结构程序为例)。

(1)输入模块 J 的源程序 800～930，先把 860 行改为“END”、800 行改为 PLAY “M0V15M1V8T3:M1V5T3:T3”。运行程序，测试 MUSIC1。纠错后，再把 800 行中的 T3 改为 T1，检查节奏是否加快。无误后，把 940 句改为 END，RUN 900↙，检查测试 MUSIC 2。无误后，删去 860 行，RUN 800，检查两程序是否能连续运行，节奏变化是否符合设计要求。无误后，测试通过。恢复 860、940 行的 RETURN，备用(注意不要清除程序)。

(2)输入模块 B 的源程序 400～480 行，删去 480 行。运行程序，检查是否能够显示出路、白云、树；成龙、金凤散步走到白云底下，秃鹰突降，然后与金凤一起斜向升空的过程是否衔接；音乐的延时是否与动作合拍。调试好后，恢复 480 行的 RETURN。

(3)输入模块 C 的源程序 500～545，把 545 行改为 END。RUN 500↙，检查能否打印出七层楼和楼梯。若能正常打印，测试通过。删去 545 行。

(4)输入模块 D 的源程序 550～615 行。把 615 行改为 END。运行 500 行，观察 9 个箱子能否都打印在楼板上，而不在楼梯和空间部分，测试通过后删去 615 行。

(5)输入模块 E 的源程序 620～685 行。把 685 行改为 END，RUN 500↙，检查能否显示楼房、箱子、成龙、金凤、妖怪等。若正常，测试通过。恢复 545、615、685 三行的 RETURN。

(6)输入模块 F 的源程序 700～785 行。添 703 行 SPRITE ON，改 785 行为 END，RUN 700↙，检查能否显示成功后的两个画面，音乐调用是否正常。测试通过后，恢复 785 行的 RETURN，

删去 703 行。

至此，6 个辅助模块全部测试完毕。下面输入主程序。

(7)输入主程序 A 的源程序 10～150 行。再添 83 行：

```
83 GOTO 75
```

运行程序，依次检查：金凤被山鹰叼走后清屏、显示 START、奏乐、清屏、显示七层楼房并提示“PRESS START”，按“START”键后，发声、显示九个箱子、成龙、金凤、妖怪，计时开始，等到 TIME＝0 时，清屏，显示“GAME OVER”和“CONTINUE/start END/select”，提示游戏结束，继续游戏按“START”键，结束游戏按“SELECT”键。先按“START”键，应能清屏，显示“START!!!”，奏乐后，重复开始过程，等待 TIME＝0 时，按“SELECT”键，应能清屏，打印“GOOD BYE!!!”后停机。

若运行正常则模块 K 测试通过。删去 83 行。

(8)输入模块 G 的源程序 200～265 行，加入 88 行：

```
88 GOTO 75
```

运行程序，检查能否用操纵器控制成龙上、下、左、右运动。正常后，结束运行，删去 88 行。

(9)输入模块 H 的源程序 270～295 行，添 93 行：

```
93 GOTO 75
```

运行程序，用操纵器控制成龙运动，走遍 9 个箱子，检查能否令一个个箱子连续消失。若箱子全部消失后，是否能进行救人成功后的处理。若正常，测试通过，删去 93 行。

(10)输入模块 I 的源程序 300～395 行，检查整个程序的运行情况。若一切正常，达到了系统功能要求，则整个程序测试通过，可以投入使用。

以上可见，增量测试法的明显优点是：

①排错较容易。因为大多数错误都与最后加进的模块有关，故确定错误位置较容易。

②对模块之间接口配合不当或模块间假设等方面的错误，可以在早期的模块组合测试时发现。而在非增量测试中，错误在结合成完整程序之前是不会发现的，而且发生了错误也难以定位。

③可以取得比较彻底的测试效果。由于在测试后一个模块时，前一个模块也随之一起执行，这样就有可能暴露出原先仅对其下层模块单独测试时的不足。

7.4.5 游戏程序的运行与维护阶段

软件通过测试、改错后，可进入运行和维护阶段。对软件的维护通常可分为三类：

1. 改正性维护：包括对软件不能达到系统说明书所要求的性能上的缺陷和编写上的隐含错误而做的改正。

2. 适当性维护：包括因适应内、外部环境的变化所需进行的变更。

3. 提高性维护：包括为了提高软件的处理效率、性能和便于维护而做的变更。

因此，重视软件的维护，可以提高软件的性能、效率与质量，延长软件的寿命。

对我们编制的程序来说，设计中尚有一些不足之处，如：妖怪运动没有声响，减少了紧张、恐怖感；成龙取到宝贝后没有击败山妖的画面；游戏主画面中的楼房设计的不够形象等等需要进行补充和变更。另外，程序中没有设计游戏难度选项，若增加游戏难度，读者可以调整 635 行中 KN 的数值，最多可增加到 7 个妖怪。还可以调整 635 行中的 TI 数值，缩短游戏时间，还可以增加藏宝箱的个数等等。读者可凭自己的兴趣取舍，不再赘述。

以上我们通过一个游戏实例较详细地介绍了游戏程序的设计过程。希读者克服急于编程的欲望，认真按照五个阶段的要求做，定能收到事半功倍的效果。

7.5 游戏程序的编程特技

通过前四节的介绍，我们已基本掌握了游戏程序的编写步骤、设计过程及动画处理技巧，能够编出一些简单的游戏程序。但玩起来总觉得不是十分满意，产生了让卡通跳起来、背景动起来、希望两人同时玩等更高的追求。实际上，富有吸引力的游戏就是因为展现了五彩缤纷、美丽漂亮的图案，产生出生动有趣、能够灵活控制的动画形象，并配以优美动听，效果逼真的音响……。为使富有想象力的朋友能够编出更新颖、更有趣的游戏程序，本节将分背景特技、卡通特技、结合特技三部分介绍游戏程序编写中的特殊技术处理。

7.5.1 背景特技

1. 字符的动态显示

若在游戏的开始，屏幕上显示出连续运动的标题字符，将使标题活泼增辉。程序 No. 7—8 可达到这一目的。

```
5 REM "No. 7—8,CAPTIONS MOVE"
10 CLS:CLEAR:N$="*CHENG LONG rescues JIN FENG*":M$
   =N$:Z$="WELCOME TO PLAY!"
20 N$=MID$(N$,2,24)+LEFT$(N$,1)
30 Z$=MID$(Z$,2,15)+LEFT$(Z$,1)
40 M$=RIGHT$(M$,1)+MID$(M$,1,25)
50 LOCATE 1,0: PRINT N$
60 LOCATE 6,10:PRINT Z$
70 LOCATE 1,20:PRINT M$
80 PAUSE 20:GOTO 20
```

运行程序，上一行循环左移显示"成龙救金凤"；下一行循环右移显示"成龙救金凤"；中间一行循环左移显示"欢迎游戏"。这犹如把字符印在电度表的铝盘上，通电后，铝盘旋转，字符循环在窗口

显示。

改变80行中PAUSE语句的参数可以调整字符移动的速度。如在程序中加入读操纵器语句

```
75 K=STRIG(0):IF K=1 GOTO 100
100 (程序的开始语句)
```

则可使用操纵器的“START”键控制游戏的开始。不按任何键，字符循环显示；按“START”键则清屏，显示游戏的开始画面。

2. 动态清屏

以往屏幕的清除都是使用CLS指令，清屏过程单调呆板。下面给出6个动态清屏程序，清屏过程如下：

(1)No. 7—9：由屏幕的上、下边开始向中间横向清屏。

(2)No. 7—10：由屏幕的中间开始向上、下边横向清屏。

(3)No. 7—11：由屏幕的中间开始向左、右边纵向清屏，犹如拉开帷幕。

(4)No. 7—12：由屏幕的左、右边开始向中间纵向清屏，犹如合上帷幕。

(5)No. 7—13：把屏幕分成四等份，交叉顺序清屏。清屏顺序如图：

1	3
4	2

(6)No. 7—14：从外向内右旋转清屏。

程序清单如下：

(1)上、下向中间清屏。

```
5 REM “No. 7—9”
10 FOR I=1 TO 21:PRINT “AAAAAAAAAAAAAAAAAAAAA
   AAAAAA”:NEXT
15 B$=“
   ”
```

```
20 FOR I=1 TO 12
25 LOCATE 0,I:PRINT B$
30 LOCATE 0,23-I:PRINT B$
35 NEXT:LOCATE 0,0
```

(2)中间向上、下清屏

```
5 REM "No. 7—10"
10 FOR I=1 TO 21:PRINT "AAAAAAAAAAAAAAAAAAAAA
   AAAAAA":NEXT
15 B$ ="
    "
20 LOCATE 0,12-I:PRINT B$
25 LOCATE 0,11+I:PRINT B$
30 NEXT:LOCATE 0,0
```

(3)从中间向两边清屏

```
5 REM "No. 7—11"
10 FOR I=1 TO 672:PRINT "C";:NEXT
20 I=1
30 FOR J=0 TO 22
40 LOCATE 14-I,J:PRINT "  "
50 LOCATE 13+I,J:PRINT "  "
60 NEXT
70 I=I+1:IF I=15 THEN 90
80 GOTO 30
90 LOCATE 0,0:END
```

(4)从两边向中间清屏

```
5 REM "No. 7—12"
10 FOR I=1 TO 672:PRINT "D";:NEXT
20 I=0
30 FOR J=0 TO 22
40 LOCATE I,J:PRINT "  "
```

```
50 LOCATE 27-I,J:PRINT "  "
60 NEXT
70 I=I+1:IF I=15 THEN 90
80 GOTO 30
90 LOCATE 0,0:END
```

(5)把屏幕分成四部分交叉清屏

```
5 REM "7—13"
10 FOR I=1 TO 672:PRINT "E";:NEXT:T=0:U=0:V=0
20 FOR Y=0 TO 11
30 LOCATE U,Y+V:PRINT "                                    "
40 NEXT:T=T+1
50 IF T=1 THEN U=14:V=11:GOTO 20
60 IF T=2 THEN V=0:GOTO 20
70 IF T=3 THEN U=0:V=11:GOTO 20
80 END
```

(6)从外向内右旋转清屏

```
5 REM "No. 7—14"
10 FOR I=1 TO 21:PRINT "FFFFFFFFFFFFFFFFFFFFFFFFF
   FFF":U=0
20 FOR I=U TO 26-U:LOCATE I,U:PRINT " ":NEXT
30 FOR J=U TO 21-U:LOCATE 27-U,J:PRINT " ":NEXT
40 FOR K=26-U TO U+1 STEP-1:LOCATE K,21-U:PRINT
   "  ":NEXT
50 FOR L=21-U TO U+1 STEP-1:LOCATE U,L:PRINT " ":
   NEXT
60 U=U+1:IF U=11 THEN LOCATE 0,0:END
70 GOTO 20
```

以上六个程序虽然短小,但清屏效果良好。程序较简单,不多说明,读者只要掌握了清屏程序的设计思路,定能编写出生动有趣的清屏程序。

3. 动态背景显示

许多电视游戏节目如《火箭车》、《机车大赛》、《南极探险》等，都采用了移动背景产生动画移动的感觉。我们也可以学习这种方法，产生动态背景效果。

设想一艘太空船要经过太空通道降落到某个星球上。我们可以设定太空船在屏幕上方不动，令太空通道和太空殒石从太空船两边飞速上移，则会产生太空船飞速下降的效果。

程序清单如下：

```
5 REM "No. 7—15,BG MOVE"
10 CLS:SPRITE ON:CGSET 0,1
15 DIM A$(14)
20 FOR I=0 TO 14
25 A$(I)=CHR$(207+I)
30 NEXT
35 DEF MOVE(0)=SPRITE(7,0,1,4,0,0)
40 POSITION 0,80,40
45 PRINT A$(RND(15)):LOCATE 17,23
50 MOVE 0:PRINT A$(RND(15))
55 GOTO 45
```

4. 背景的显示、消失与重显

在《采金块》的游戏中，小熊用手一指地面，即可在地面上挖出一个小坑，埋掉追赶它的敌人。这就是使用了背景画面的消失与重显技术。实现的方法是在欲消失背景的位置上打印空格，(即用空格覆盖背景)打印空格前要先用串变量记忆被覆盖的背景，恢复显示时，再把记忆的背景重新打印出来。

实现方法及示范程序见 No. 7--16

```
5 REM "No. 7—16,BG ELIMINATE and RESHOW"
10 CLS:SPRITE ON
15 FOR I=0 TO 26 STEP 2
```

```
20 X=I:Y=15:GOSUB 65
25 X=I:Y=17:GOSUB 65
30 NEXT
35 DEF MOVE(7)=SPRITE(0,3,1,1,0,0):POSITION 7,124,128:
   MOVE 7
40 S=0:S=STRIG(0)
45 X=(XPOS(7)-12)/8:Y=(YPOS(7)-16)/8
50 IF S=4 THEN X=X-2:GOSUB 80
55 IF S=8 THEN X=X+2:GOSUB 80
60 GOTO 40
65 LOCATE X+1,Y:PRINT CHR$(247):LOCATE X,Y:PRINT
   CHR$(243)
70 LOCATE X,Y+1:PRINT CHR$(245):LOCATE X+1,Y+1:
   PRINT CHR$(248)
75 RETURN
80 A$=SCR$(X,Y+1):B$=SCR$(X+1,Y+1)
85 C$=SCR$(X,Y+2):D$=SCR$(X+1,Y+2)
90 LOCATE X,Y+1:PRINT "  "
95 LOCATE X,Y+2:PRINT "  "
100 PAUSE 100
105 LOCATE X,Y+1:PRINT A$;B$
110 LOCATE X,Y+2:PRINT C$;D$
115 RETURN
```

程序说明：

①10～35 行为初始化程序，显示地面和玛丽。

②40～60 行为主控程序。

③65～75 行为画地面子程序。

④80～115 行为地面消失和重显子程序。其中由 80、85 两行记录消失前的背景画面。

运行程序 No. 7—16，使用 I″操纵器的 A、B 键可控制玛丽在

左、右挖坑。挖坑后延时一段时间自动填平。

若使用在程序中，可把 100 行改为 RETURN，(返回调用程序)，在主控程序中加入一延时控制语句，如

D=D+1:IF D>100 THEN GOSUB 105

5. 用操纵器控制闪烁光标在屏幕上移动

在中华学习机的《挖金砖》游戏中，游戏者可使用键盘控制一闪烁光标移动，与金砖重合后挖取金砖。在我们的学习机中只有用操纵器控制卡通运动的示例，而没有介绍如何使用操纵器控制光标运动。但这种控制形式在游戏程序中经常用到。如在游戏的选项中，用操纵器的方向键控制光标选项等等。

现在，我们讨论控制光标的方法。

(1)闪烁光标的实现

中华学习机中的闪烁光标是通过对某一固定点交替染黑、白颜色实现的。在我们的机器中也可以采用这一方法。先使用 PALETB 语句定义黑、白色块代码；然后在显示光标的位置打印某种背景图案，如 M8 等；再使用 COLOR 语句进行染色，从而达到显示闪烁光标的目的。但是，在使用中发现这种方法获得的闪烁光标不理想。它不仅显示的光标块太大(由四个低分辨图形象素构成)而且还要占用两个色块代码，使用也不方便。

获得闪烁光标的另一方法是：在闪烁光标的座标处直接打印 M6 号背景图案(若一块不够长，可使用两块)；延时一段时间后，打印空格进行覆盖；再延时一段时间后，再打印 M6 号图案。这样构成的闪烁光标，大、小可随意调节，式样可任意选择，使用灵活方便。

(2)闪烁光标的控制

闪烁光标确定以后，可通过读操纵器的方向键值而改变光标显示的座标，即可达到控制光标移动的目的。示范程序见 No. 7—17：

```
5 REM "No. 7—17;CURSOR MOVE MENT"
10 CLS:X=10:Y=10:A$=CHR$(253)
20 K=0:SX=0:SY=0
30 K=STICK(0):IF K=0 THEN 80
40 IF K=1 THEN SX=2
50 IF K=2 THEN SX=-2
60 IF K=4 THEN SY=1
70 IF K=8 THEN SY=-1
80 X=X+SX:IF X>27 THEN X=26
90 IF X<0 THEN X=0
100 Y=Y+SY:IF Y>20 THEN Y=20
110 IF Y<6 THEN Y=6
120 GOSUB 130:GOTO 20
130 LOCATE X,Y:PRINT A$;A$:PAUSE 30
140 LOCATE X,Y:PRINT "    ":PAUSE 15
150 RETURN
```

程序说明：

①20～120 行为光标移动程序

②130～150 行为光标闪烁程序

程序中通过 80～110 行限制了光标的移动范围在(0,6)→(0,20)→(26,6)→(26,20)之间。若欲扩大、缩小移动范围，可调整这四行语句中 X、Y 的赋值。

6. 背景的换场技术

多数电视游戏的背景画面都要进行换场。如《爱的小屋》、《采金块》、《坦克》、《爆破手》等游戏，每一关都要换一个游戏场面，从而使游戏的难度增大，技巧提高。可以说，除棋、牌类游戏外，老是固定一个场面的游戏肯定枯燥乏味。但是用 F BASIC 语言编制游戏程序与任天堂游戏卡的游戏程序截然不同。因为游戏卡内有两套存储器分别储存游戏管理程序和图案信息程序。程序运行时，由

CPU 处理游戏管理程序，由 PPU 处理图案信息程序，它们都是用机器语言编写的，不须解释，可直接执行，所以处理速度快、画面精彩多变。与其相比，使用 F BASIC 语言编程则望尘莫及。但是，我们可以在现有情况下，最大限度地发挥它的作用。只要精心设计背景图案，就可弥补游戏换场速度较慢的不足。

那么如何实现换场呢？这是一个较复杂的问题。现在仅介绍一种简单换场的方法。

我们知道游戏学习机有三种显示方式，即

①文本方式：28 字×24 行，字符结构为 8×8 点阵。

②低分辨彩色图形方式：分辨率为 20 行×28 列。

③高分辨彩色图形方式：分辨率为 256 行×240 列。

据我研究发现，在低分辨彩色图形方式下存在两个显示页面。在一个页面显示的同时，可在另一个页面绘图。开机后，系统自动设定为三种方式共显，通过 SCREEN 语句可以选择显示方式。在背景的换场操作中，使用 SCREEN 语句的方法如下：

(1)SCREEN 0,1

这时屏幕上仍显示前一场内容。若转入绘图程序，可把后一场图案绘制出来。因而可实现前台显示，后台绘图(即无显示绘图)。

(2)SCREEN 0,0:VIEW

这时，可把幕后绘制的后一场的图案显示出来。即显示第二图形页的图案。

(3)再使用“SCREEN 0,1”，又可绘制下一场的内容，即在第一图形页绘图。

(4)再使用“SCREEN 0,0:VIEW”，又可显示第一图形页的内容。

提请注意的是：各图形的内容送往显示缓冲区时没有清屏作用。因此，在显示第一页时，若又送入第二页的内容，则两页的图形是相加的关系，即第二页图形的空格部分会保留第一页的图形。但

是，两图形页显示的中间又不能使用清屏命令“CLS”（因 CLS 可同时清除两个显示页），否则将把两场图形都清掉。因此必须采取另外的清屏方法。有效的方法是编程清屏。清屏程序如下：

```
200 FOR I=0 TO 23
210 PRINT "
            "
220 NEXT
```

解决了清屏问题后，就可进行背景的换场操作了。示范程序见 No. 7—18

```
5 REM "No. 7--18,BG CHANGE"
10 SCREEN 0,1
30 FOR I=0 TO 500
40 PRINT CHR$(199);:NEXT
50 SCREEN 0,0:VIEW
60 SCREEN 0,1
80 FOR I=0 TO 400
90 PRINT CHR$(207);:NEXT
100 SCREEN 0,0:VIEW
110 SCREEN 0,1
120 FOR I=0 TO 200
130 PRINT CHR$(216);CHR$(217);CHR$(218);:NEXT
140 SCREEN 0,0:VIEW
150 GOTO 10
160 FOR I=0 TO 23
170 PRINT "……"(28 个空格)
180 NEXT
190 RETURN
```

程序中 10～40 行完成在屏幕上打印 501 个小旗的图案。（幕后绘图）

50 行令其显示在屏幕上

60～90 行完成幕后打印 401 个苹果。

100 行令其显示。

110～130 行完成幕后绘制 201 朵白云的图案。

140 行令其显示。

运行程序 No. 7—18。首先显示 501 个小旗；延时一段时间后显示 401 个苹果，下面留下了 100 个小旗没有被覆盖；又延时一段时间，显示 201 朵白云覆盖了苹果和小旗。以后又显示 501 个小旗，下面留下了 33 朵白云；再显示 401 个苹果，下面留下了 100 个小旗和 33 朵白云。可见各场之间只能覆盖、不能清屏。

按“Ctrl＋Alt＋Del”键中止程序运行（这时按“Stop”键已不起作用），加入如下两行程序：

```
20 GOSUB 160:LOCATE 0,0
70 GOSUB 160:LOCATE 0,0
```

再运行程序。现在，每场或者显示 501 个小旗，或者显示 401 个苹果，或者显示 201 朵白云，不再有混显现象。

这一示范程序，打通了我们在游戏程序中换场的思路。即在第一关打通后，先保留第一关的画面，立即绘制第二关的画面。绘图完成后进行程序清屏，随后显示第二关的画面，进行第二关游戏。以后各关都采用此方法，即可达到游戏换场的目的。

显而易见，在程序中应用换场技术是行之有效的。

7.5.2 卡通特技

1. 卡通的随机运动

关于卡通的随机运动，我们在《企鹅看外婆》游戏程序中已经涉及，现予以系统介绍。

卡通的随机运动有七种形式：第一，卡通的种类随机选定；第二，运动的方向随机确定；第三，运动的速度随机决定；第四，运动的距离随机选定；第五，卡通的颜色随机认定；第六，卡通显示于背景面的优先度随机指定；第七，卡通运动的起点位置随机确定。至

于选择几种随机形式，一般根据游戏的需要而灵活掌握。例如在《企鹅看外婆》游戏中，我们仅选用了汽车的颜色和运动起点座标两种随机形式。

程序 No. 7—19 演示了七种形式的综合表现：

```
5 REM "No. 7—19,SP. RANDOM MOVE"
10 CLS:SPRITE ON
15 PALETB 0,33,32,38,28
20 FOR I=0 TO 55
25 PRINT CHR$(216);CHR$(217);CHR$(218);
30 NEXT
35 LOCATE 0,15
40 FOR I=0 TO 223
45 PRINT CHR$(255);:NEXT
50 FOR I=0 TO 7:GOSUB 75:NEXT
55 FOR I=0 TO 7
60 IF MOVE(I)=0 THEN GOSUB 75
65 NEXT
70 GOTO 55
75 DEF MOVE(I)=SPRITE(RND(16),RND(8)+1,RND(9)+1,
   RND(155)+100,RND(2),RND(4))
80 POSITION I,RND(236)+20,RND(240):MOVE I
85 RETURN
```

运行程序，在天蓝色的背景上，上面一片白云，下面一块绿地，16 种卡通以不同的色彩，不同的速度，不同的运动方向和运动起点穿梭运动于兰天下、绿地上、白云中，构成一幅绚丽的画卷。读者可从中深刻体会到七种随机运动形式综合作用的奇妙。

2. 卡通的自动运动

所谓卡通的自动运动，是指不用游戏者操纵卡通自动进行的运动。实质上是由程序控制的运动。卡通的自动运动一般分为两大类：一类是无目的的运动，如程序 No. 7—19 中的运动；一类是

有目的运动，如《企鹅看外婆》游戏中汽车运动的目的是阻碍企鹅太太转移小企鹅，《成龙救金凤》游戏中妖怪的运动在于干扰成龙打开宝箱。一般无目的运动容易实现，而有目的运动则要复杂得多。《成龙救金凤》游戏中用了较大的程序量才实现了妖怪自动向成龙靠近的运动控制。卡通的有目的的自动运动是游戏程序中最常用的特技之一。请读者仔细阅读程序 No. 7—6 中模块 I 的源程序，以加深印象。

3. 用操纵器控制卡通跳跃

卡通的跳跃有两种情况：一是原地跳；一是运动中跳。运动中跳还分向左跳、向右跳。我们知道，F BASIC 语言中没有令卡通跳跃的语句，而 MOVE 语句只能令卡通在确定的位置开始运动，也不能产生跳跃。因此，只能通过 SPRITE 语句，以不同的时间间隔、在不同的位置、显示卡通不同的姿式而产生卡通跳跃的动作。

设计卡通跳跃的基本思路如下(以玛丽为例)：

我们知道在卡通图形库中有玛丽走、跳、跌、滑等七种运动姿式，若再通过 DEF SPRITE 语句中 X 轴、Y 轴的翻转可得到 28 种运动姿式。在设计运动卡通跳跃时，开始先显示玛丽走的姿式，随后在跳高座标上显示跳的姿式，最后在落地座标上再显示走的姿式，(同时加入 X 方向的位移，使其跳跃后向前移动一段距离。)则完成了一次运动跳跃。若是原地跳，则在跳起前显示站立的姿式，在跳高座标上显示跳的姿式，随后显示跌下的姿式，则完成了一次原地跳跃。各姿式之间的衔接关系是：先显示后一姿式，再消失前一姿式，延时一段时间，再显示下一姿式，消失前一姿式。

具体的实现方法是：定义 0# 显示卡通为跌下姿式，1# 为面向右跳姿式，2# 为面向左跳姿式，7# 为运动卡通，选用操纵器的 A 键控制跳跃。现在假定玛丽在向右运动中按 A 键，则跳跃过程如下：

①SP 1,X1,Y1　　(在跳高座标上显示 1# 姿式)

②ERA 7　　(令 7# 动作消失)

③PAUSE 5　　（延时）

④POSITION 7,X2,Y2:MOVE 7　（在新的座标上令玛丽显示并运动）

⑤SPRITE 1　　（令 1# 显示卡通消失）

这样就实现了卡通向右运动中的一次跳跃。同理，向左运动时，在跳高座标上显示 2# 显示卡通；原地跳时则显示 0# 显示卡通。演示程序见 No. 7—20。

```
5 REM "No. 7—20,SP. JUMP"
10 CLS:SPRITE ON:X=100:Y=100
15 DEF SPRITE 0,(0,1,0,1,0)=CHR $ (25)+CHR $ (24)+CHR
   $ (27)+CHR $ (26)
20 DEF SPRITE 1,(0,1,0,1,0)=CHR $ (13)+CHR $ (12)+CHR
   $ (15)+CHR $ (14)
25 DEF SPRITE 2,(0,1,0,0,0)=CHR $ (12)+CHR $ (13)+CHR
   $ (14)+CHR $ (15)
30 DEF MOVE(7)=SPRITE(0,V,1,4):POSITION 7,X,Y:MOVE
   7:SPRITE S
35 S=STICK(0):IF(S=1)+(S=2)=0 THEN S=0:V=0
40 K=STRIG(0):IF K=8 THEN 60
45 IF S=1 THEN V=3
50 IF S=2 THEN V=7
55 X=XPOS(7):Y=YPOS(7):GOTO 30
60 X=X+6*((S=2)-(S=1)):IF X<12 THEN X=240
65 IF X>240 THEN X=12
70 Y=Y+20*((K=8)-(F=1))
80 IF F=0 THEN SPRITE S,X,Y:ERA 7:PAUSE 5:F=1:K=0:
   GOTO 60
85 POSITION 7,X,Y:F=0:V=2:GOTO 30
```

程序说明：

①35 行的 IF—THEN 语句的作用是封锁操纵器除左、右运动外的 6 个方向键。

②60 行产生 X 方向的位移，70 行产生 Y 方向的位移。两语句中均使用了运算符号判别式，其作用如下：

在 X 轴方向，当卡通向右运动时，有 S＝1，那么有(S＝2)＝0、(S＝1)＝－1、则 X＝X＋6，故卡通跳起时向前移动 6 点，落地时再向前移动 6 点，一次跳跃在 X 方向上共向前移动 12 点。卡通的运动轨迹是等腰三角形的两条边和顶点组成的折线。同理，当卡通向左运动时有 S＝2，则有(S＝2)＝－1、(S＝1)＝0、X＝X－6，一次向左跳跃也向前移动 12 点。

在 Y 轴方向，若按 A 键则 K＝8，这时 F＝0，那么 Y＝Y－20，则在(X，Y－20)的座标上显示跳的姿式(相当于跳起)，随后 K＝0，F＝1，再返回 70 行时则有(K＝8)＝0、(F＝1)＝－1、Y＝Y－20＋20＝Y，纵向座标又返回到跳起前的数值(相当于卡通落下)。

4. 用操纵器控制卡通运动的速度

在《超级玛丽》的游戏中按 B 键可令玛丽的运动速度增加一倍，在《火箭车》游戏中按 B 键也可提高车速。这一过程是如何实现的呢？说透了，很简单。方法是：在定义受操纵的卡通时，用一变量 D 代替它的速度值。正常情况下 D＝2，当读到 STRIG(0)＝4 时(即按 B 键)则令 D＝1，从而速度增加一倍。若松开 B 键仍然 D＝2，维持原速。这样就实现了速度的控制。在程序 No. 7—20 中改 10、30 行、添 43 行：

```
10 CLS:SPRITE ON:X=100:Y=100:D=2
30 DEF MOVE(7)=SPRITE(0,V,D,4):POSITION 7,X,Y:MOVE
   7:SPRITE S
43 D=2:IF K=4 THEN D=1
```

运行程序，按 B 键就可使玛丽的运动速度提高一倍。

5. 双操纵器的使用

《魂斗罗》、《双截龙》、《忍者》等游戏都是双打节目，游戏中使用两个操纵器分别控制两个主人翁的动作。由于战斗人员的增加

出现了战斗的配合和任务的分工，从而使游戏更加激烈、有趣、富有吸引力。

有了单操纵器的使用经验和方法，双操纵器的使用也变得很简单。一般来说，只要编写两个单操纵器的控制程序，令Ⅰ#操纵器控制一个卡通（例如1#），令Ⅱ#操纵器控制另一个卡通（如2#）。然后将两个程序连接在一起，使其在程序运行中先测试Ⅰ#操纵器，执行1#卡通的动作控制后，再测试Ⅱ#操纵器执行2#卡通的动作控制。这样对两个操纵器顺序测试、对两个卡通顺序控制，就达到了双操纵器分别控制的要求。程序No. 7—21为双操纵器控制玛丽、莉莎动作的示范程序，由此可窥一斑。

```
5 REM "No. 7—21"
10 CLS:SPRITE ON
15 FOR I=0 TO 1
20 DEF MOVE(I)=SPRITE(I,0,1,8):POSITION I,100,I*80+80:
   MOVE I:NEXT
25 S=0:V=0:S=STICK(0):IF S=0 THEN 70
30 IF S=1 THEN V=3
35 IF S=2 THEN V=7
40 IF S=4 THEN V=5
50 IF S=8 THEN V=1
55 X0=XPOS(0):Y0=YPOS(0)
60 DEF MOVE(0)=SPRITE(0,V,1,8):POSITION 0,X0,Y0:MOVE
   0
70 S=0:V=0:S=STICK(1):IF S=0 THEN 25
75 IF S=1 THEN V=3
80 IF S=2 THEN V=7
85 IF S=4 THEN V=5
90 IF S=8 THEN V=1
95 X1=XPOS(1):Y1=YPOS(1)
100 DEF MOVE(1)=SPRITE(1,V,1,8):POSITION 1,X1,Y1:
```

```
    MOVE 1
105 GOTO 25
```

读者可能已经发现，程序 No. 7—21 太罗索。实用中，对两操纵器的测试和卡通的控制是通过一个转换变量进行的。见程序 No. 7—22

```
5 REM "No. 7—22"
10 CLS:SPRITE ON
15 FOR I=0 TO 1
20 DEF MOVE(I)=SPRITE(I,0,1,8):POSITION I,100,I*80+80:
   MOVE I:NEXT
25 F=F+1:IF F>1 THEN F=0
30 S=0:V=0:S=STICK(F):IF S=0 THEN 25
35 IF S=1 THEN V=3
40 IF S=2 THEN V=7
45 IF S=4 THEN V=5
50 IF S=8 THEN V=1
55 X=XPOS(F):Y=YPOS(F)
60 DEF MOVE(F)=SPRITE(F,V,1,8):POSITION F,X,Y:MOVE
   F
65 GOTO 25
```

程序中使用了变量 F 为两操纵器、卡通的转换变量，从而使程序简洁精炼。

6. 卡通游泳

画一水面，令卡通在背景后面缓慢运动，并使其头部露出水面，这样就形成了卡通游泳的动画。实现程序见 No. 7—23

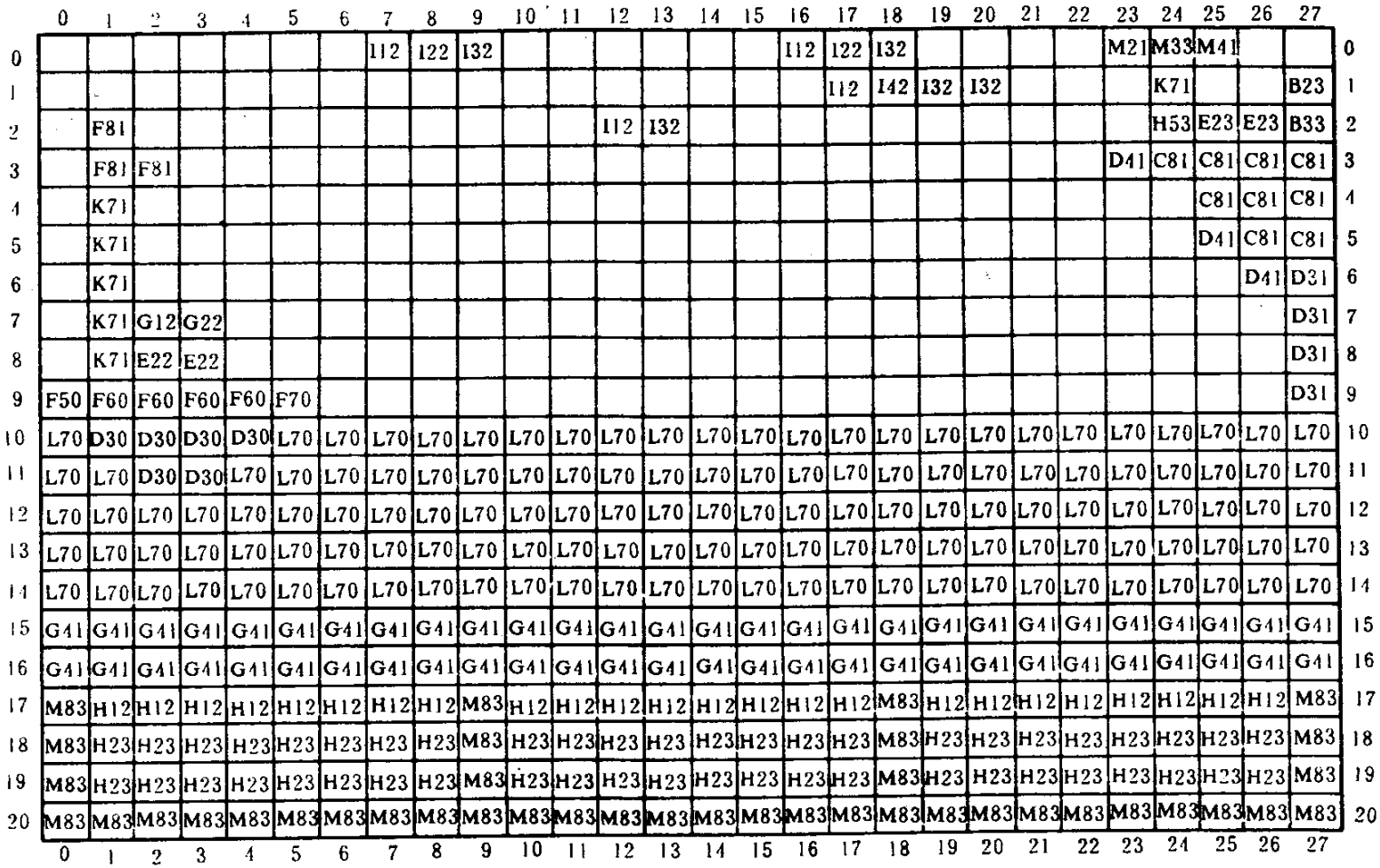
```
5 REM "No. 7—23"
10 CLS:SPRITE ON
20 LOCATE 0,10:FOR I=0 TO 140:PRINT CHR$(254);:NEXT
30 FOR I=0 TO 140:PRINT CHR$(202);:NEXT
40 LOCATE 6,0:PRINT CHR$(216);CHR$(217);CHR$(218)
```

```
50 LOCATE 7,1:PRINT CHR $ (216);CHR $ (217);CHR $ (217);
   CHR $ (218)
60 DEF MOVE(0)=SPRITE(0,3,4,100,1,1):POSITION 0,20,95:
   MOVE 0
```

运行程序，玛丽哥哥在水中从屏幕左边游到右边。

若欲使用操纵器控制卡通游泳，则可使其潜入水中，跃出水面，从水中走出在沙滩上奔跑，再走入水中继续游泳等等。欲使画面生动形象，可键入 BGTOOL 命令进入 BG 绘图画面（使用 BS 卡的键盘可键入 SYSTEM 命令调出选择菜单，键入 2 进入 BG 画面）按图 7—25 进行键控绘图。绘图后键入程序 No. 7—24。

```
5 REM "No. 7—24"
10 VIEW:SPRITE ON
15 DEF MOVE(0)=SPRITE(0,0,4,4,1,0):POSITION 0,50,96:
   MOVE 0
20 S=0:S=STICK(0):IF S=0 THEN 75
25 IF S=1 THEN V=3
30 IF S=2 THEN V=7
35 IF S=8 THEN V=1
40 IF S=4 THEN V=5
45 X=(XPOS(0)-12)/8:IF X>25 THEN X=4
50 IF X<4 THEN X=25
55 Y=(YPOS(0)-16)/8:IF Y<10 THEN Y=10
58 IF Y>15 THEN Y=15
60 IF Y>13 THEN 70
65 DEF MOVE(0)=SPRITE(0,V,3,4,1,0):POSITION 0,X * 8+
   12,Y * 8+16:MOVE 0:GOTO 75
70 DEF MOVE(0)=SPRITE(0,V,1,4,0,0):POSITION 0,X * 8+
   12,Y * 8+16:MOVE 0
75 PLAY "T1C1D#E#F#GCDEF#G":GOTO 20
```



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	
0								I12	I22	I32							I12	I22	I32					M21	M33	M41			0
1																		I12	I42	I32	I32				K71			B23	1
2		F81											I12	I32											H53	E23	E23	B33	2
3		F81	F81																					D41	C81	C81	C81	C81	3
4		K71																								C81	C81	C81	4
5		K71																								D41	C81	C81	5
6		K71																									D41	D31	6
7		K71	G12	G22																								D31	7
8		K71	E22	E22																								D31	8
9	F50	F60	F60	F60	F60	F70																						D31	9
10	L70	D30	D30	D30	D30	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	10
11	L70	L70	D30	D30	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	11
12	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	12
13	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	13
14	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	L70	14
15	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	15
16	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	G41	16
17	M83	H12	H12	H12	H12	H12	H12	H12	H12	M83	H12	H12	H12	H12	H12	H12	H12	H12	M83	H12	H12	H12	H12	H12	H12	H12	H12	M83	17
18	M83	H23	H23	H23	H23	H23	H23	H23	H23	M83	H23	H23	H23	H23	H23	H23	H23	H23	M83	H23	H23	H23	H23	H23	H23	H23	H23	M83	18
19	M83	H23	H23	H23	H23	H23	H23	H23	H23	M83	H23	H23	H23	H23	H23	H23	H23	H23	M83	H23	H23	H23	H23	H23	H23	H23	H23	M83	19
20	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	M83	20
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	

图 7—25　玛丽游泳绘图规则

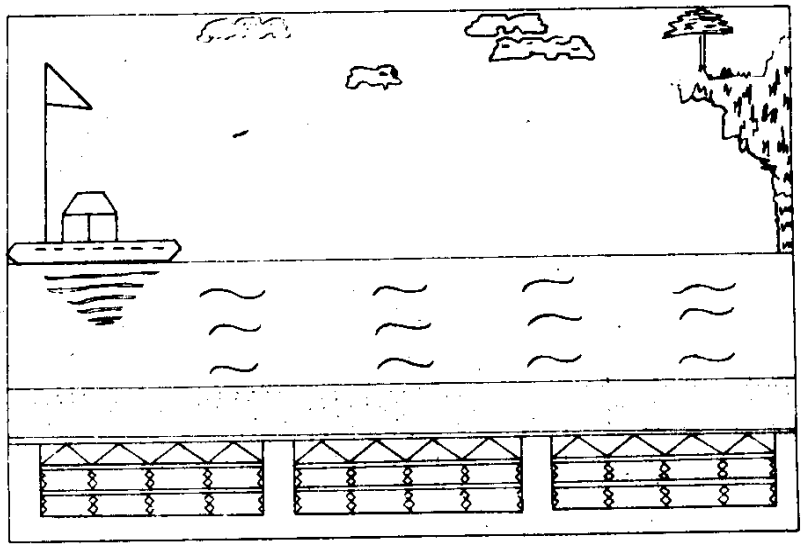


图 7—26　玛丽游泳画面

程序中 70 行的作用是：当卡通潜水到水边沙滩附近时，使其显示于背景前面，并可在沙滩上奔跑。读者若喜欢卡通跳跃，可在 20 行之后加入跳跃控制语句。

运行程序，用操纵器的左、右键控制玛丽在水中游泳；按向下键可令其潜入水中；继续按向下键可令卡通在沙滩上显示；按左、右键可控制玛丽在沙滩上奔跑；按向上键又可令其走入水中继续游泳。整个过程极为逼真。

7. 运动卡通的放大与缩小

在《超级玛丽》游戏中，玛丽吃到红蘑菇后，身体立即长大一倍。对这种变化，许多朋友感到很奇怪，这种身体大、小的变化是如何实现的呢？现在讨论实现的方法。

我们知道，在立体分解画面指令中有一条 DEF SPRITE 语句。使用该语句，通过变量 B 可以定义显示在 SP 面上的卡通图案发生身体大小的变化。当 B＝1 时，卡通为 16×16 点，即有四个字符大小；当 B＝0 时，卡通为 8×8 点，只有一个字符大小。只要解决了用操纵器控制卡通运动的问题，就可实现运动卡通的放大与缩小。实现程序如下：

```
5 REM "No. 7—25"
10 CLS:SPRITE ON:CGSET 1,0:PALETB 0,41,48,48,48
20 DEF SPRITE 0,(0,0,0,1,0)=CHR$(29)+CHR$(28)+CHR
   $(31)+CHR$(30)
30 DEF SPRITE 1,(0,0,0,1,0)=CHR$(33)+CHR$(32)+CHR
   $(35)+CHR$(34)
40 DEF SPRITE 2,(0,0,0,1,0)=CHR$(37)+CHR$(36)+CHR
   $(39)+CHR$(38)
50 DEF SPRITE 3,(0,1,0,1,0)=CHR$(29)+CHR$(28)+CHR
   $(31)+CHR$(30)
60 DEF SPRITE 4,(0,1,0,1,0)=CHR$(33)+CHR$(32)+CHR
   $(35)+CHR$(34)
70 DEF SPRITE 5,(0,1,0,1,0)=CHR$(37)+CHR$(36)+CHR
   $(39)+CHR$(38)
80 X=0:Y=100
90 X=X+9:IF X>250 THEN 140
100 SPRITE 0,X,Y:SPRITE 3:PAUSE 20
110 SPRITE 1,X+3,Y:SPRITE 0:PAUSE 20
120 SPRITE 2,X+6,Y:SPRITE 1:PAUSE 20
130 GOTO 90
140 X=0
150 X=X+9:IF X>250 THEN 80
160 SPRITE 3,X,Y:SPRITE 5:PAUSE 20
170 SPRITE 4,X+3,Y:SPRITE 3:PAUSE 20
180 SPRITE 5,X+6,Y:SPRITE 4:PAUSE 20
190 GOTO 150
```

程序中20～40行定义三个8×8点阵的SP；50～70行定义三个16×16点阵的SP；80～130行控制小SP运动；140～190行控制大SP运动。当小SP运动250点后转为大SP运动；大SP运动250点后又转为小SP运动。如此周而复始，反复演示。

程序 No. 7—25 中重复语句行较多，可通过循环语句使程序简化。简化程序见 No. 7—26。

```
5 REM "No. 7—26"
10 CLS:SPRITE ON:CGSET 1,0:PALETB 0,41,48,48,48
20 FOR I=0 TO 2:FOR J=0 TO 3 STEP 3
30 DEF SPRITE I+J,(0,-(J=3),0,1,0)=CHR$(29+I*4)+CHR
   $(28+I*4)+CHR$(31+I*4)+CHR$(30+I*4)
40 NEXT:NEXT
50 X=0:Y=100:F=0:A=0
60 X=X+9:IF X>250 AND F=0 THEN X=5:SPRITE A+2:A=3:
   F=1
70 IF X>250 AND F=1 THEN X=5:SPRITE A+2:A=0:F=0
80 SPRITE A,X,Y:SPRITE A+2:PAUSE 20:SPRITE A+1,X+3,
   Y:SPRITE A:PAUSE 20:SPRITE A+2,X+6,Y:SPRITE A+1:
   PAUSE 20:GOTO 60
```

8. 自己设计卡通

在编制游戏程序中，我们往往不满意系统给定的 16 种卡通图案，想自己创造新的卡通以开拓更新更广的游戏境界。第四章中，我们曾讨论了使用 CGEN 语句把卡通图案作为字符随意组合成各种图案的方法。但是，用这种方法组合的卡通图案只能显示不能移动。现在我们讨论如何令自己设计的卡通图案运动，如何使用操纵器控制其运动的方法。

我们在介绍 DEF SPRITE 语句时曾谈到：每一个卡通图案都由一组数字所代表，这些数字称为代码(一般四个代码为一组)，只要选定一组代码就可定义一个卡通图案。每个代码的含义就是把一个卡通图案分为四个象元，每一个象元用一个代码表示(见符号表 A)。也就是说，所有卡通图案都是由不同数量的不同象元构成的，选用不同的象元组合，可以构成千变万化的卡通图案。特别是再加入 X 轴、Y 轴反转的作用，构成的 SP 将更加妙趣横生、妙不

可言。例如：选用“8，65，10，11”可组成长翅膀的玛丽；选用“40，41，186，187”可组成丽莎坐在乌龟背上的图案；选用“144、145、174、175”可组成新的飞行器等等。程序 No. 7—27 演示了把玛丽进行 X 轴、Y 轴反转后组成的“三飞鸟”卡通运动的实现方法。

```
5 REM "No. 7—27"
10 CLS:SPRITE ON:DIM A$(3):CGSET 0,0:PALETB$ 0,33,48,
   48,48
20 FOR I=1 TO 3:C=I*4
30 A$(I)=CHR$(C-1)+CHR$(C-4)+CHR$(C-2)+CHR
   $(C-3)
40 DEF SPRITE I-1,(0,1,0,1,1)=A$(I)
50 NEXT
60 A=A+1:IF A>3 THEN A=0
70 B=A-1-(A=0)*4
80 X=X+1:IF X>255 THEN X=0
90 SPRITE A,X,100:SPRITE B:PAUSE 2:GOTO 60
```

运行程序可看到一个新的卡通——“三飞鸟”从屏幕左边运动到右边，反复运行不止。三飞鸟的形状是：一只大鸟在上边，两只小鸟隔开较大的距离在下边，犹如飞鸟妈妈在教两个孩子练飞行。

在程序 No. 7—27 中只要加入操纵器控制语句就可以使用操纵器控制“三飞鸟”运动。见程序 No. 7—28：

```
5 REM "No. 7—28"
10 CLS:SPRITE ON:DIM A$(3):CGSET 0,0:X=100:Y=100:
   PALETB 0,33,48,48,48
20 FOR I=1 TO 3:C=I*4
30 A$(I)=CHR$(C-1)+CHR$(C-4)+CHR$(C-2)+CHR
   $(C-3)
40 DEF SPRITE I-1,(0,1,0,1,1)=A$(I)
50 NEXT
60 A=A+1:IF A>3 THEN A=0
```

```
70 B=A-1-(A=0)*4
80 S=0:S=STICK(0):IF S=0 THEN 130
90 IF S=1 THEN X=X+1:IF X>255 THEN X=0
100 IF S=2 THEN X=X-1:IF X<0 THEN X=255
110 IF S=8 THEN Y=Y-1:IF Y<0 THEN Y=240
120 IF S=4 THEN Y=Y+1:IF Y>240 THEN Y=0
130 SPRITE A,X,Y:SPRITE B:PAUSE 2:GOTO 60
```

程序中的60～130行完成了用操纵器控制三飞鸟的运动。

运行程序，在屏幕中央处显示三飞鸟在原地拍翅待命。按动操纵器的方向键，可使三飞鸟向前、后、上、下飞行。

下面向读者介绍一个智能类的游戏程序——《跳马》。

例 7.5 《跳马》是两人对战的跳棋游戏，游戏双方各将一枚棋子放入8×8格的棋盘中，以中国象棋中跳马的方式走棋。所不同的是：象棋中的马是以“日”字对角为路线，从一个交叉点走到另一个交叉点；而游戏中的跳马虽仍遵循以“日”字对角为走棋路线的规则，却是从一个空格走到另一个空格，且没有“别马腿”的限制。各个方向的走棋规则见图7—26：

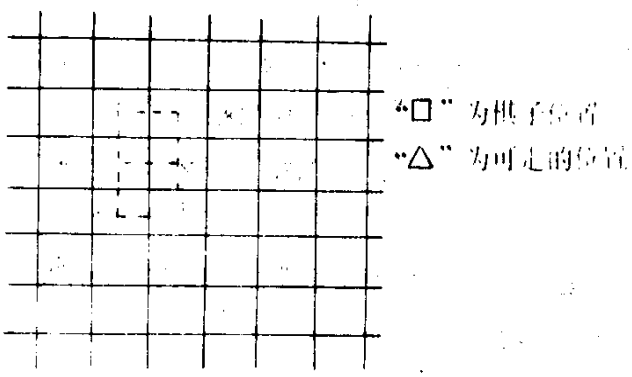


图7—26 《跳马》走棋规则

若把各空格的中心用直线连接（见图中虚线）则与象棋中的跳马走法完全相同。

从图7—26可见，每一个棋子共有八个点可走。游戏规定：棋盘中每个空格只能走一次，凡走过的空格即为禁地，不得再走（游

戏中凡棋子走过的地方均打印“＊”号），最终无路可走者为输棋。

该游戏程序可在所有F BASIC版本系统中运行。程序清单如下：（见程序No. 7—29）

```
5 REM "No. 7—29,KNIGHT"
10 VIEW:CGEN 3:CGSET 1,1
20 DEF SPRITE 0,(0,0,0,0,0)=CHR$(199)
30 PALETS 0,13,22,39,2
40 DEF SPRITE 2,(0,0,0,0,0)=CHR$(215)
50 DEF SPRITE 1,(0,0,0,0,0)=CHR$(199)
60 PALETS 1,13,22,23,4
70 DEF SPRITE 3,(1,0,0,0,0)=CHR$(215)
80 SPRITE ON
90 DIM HX(1),HY(1),X(7),Y(7),B(7,7)
100 X(0)=-1:X(1)=-2:X(2)=-2:X(3)=-1:X(4)=1:X(5)=
    2:X(6)=2:X(7)=1
110 Y(0)=-2:Y(1)=-1:Y(2)=1:Y(3)=2:Y(4)=2:Y(5)=1:Y
    (6)=-1:Y(7)=-2
190 C=0:GOSUB 250
200 C=1:GOSUB 250
210 C=1+(C=1)
220 GOSUB 390
230 IF F=-1 THEN 560
240 GOTO 210
250 X=0:Y=0:F=0
260 GOSUB 440
270 IF F=1 THEN PLAY "T1O3C2":F=0:GOTO 260
280 IF T=8 THEN RETURN
290 IF S=4 THEN Y=Y+1:IF Y>7 THEN Y=7
300 IF S=8 THEN Y=Y-1:IF Y<0 THEN Y=0
310 X=X+(S=1)-(S=2):X=-X*(X>0)+(X>7)
```

```
320 GOTO 260
340 GOSUB 440:F=0
350 IF T=8 THEN RETURN
360 IF S=0 THEN S=4
370 IF S=8 THEN N=N-1:IF N<0 THEN N=7
380 IF S=4 THEN N=N+1:IF N>7 THEN N=0
390 X=HX(C)+X(N):Y=HY(C)+Y(N)
400 F=F+1:IF F>8 THEN F=-1:RETURN
410 IF X<0 OR X>7 OR Y<0 OR Y>7 THEN 360
420 IF B(X,Y)=1 THEN 360
430 GOTO 340
440 SPRITE C,136-16*X,16*Y+47
450 T=STRIG(C):S=STICK(C)
460 IF(S+T)=0 THEN 450
470 IF T<>8 THEN 540
480 IF B(X,Y)=1 THEN F=1:RETURN
490 B(X,Y)=1
500 HX(C)=X:HY(C)=Y
510 SPRITE C+2,136-HX(C)*16,16*HY(C)+47
520 LOCATE 15-2*HX(C),3+2*HY(C)
530 PRINT "*":PLAY "T1O3CDEG"
540 SPRITE C
550 RETURN
560 REM "END ROUTINE"
570 LOCATE 3,20:PLAY "T1O3CDET2O4EGAC"
580 IF C=1 THEN PRINT "BLUE"
590 IF C=0 THEN PRINT "RED"
600 PRINT "WIN!!!":END
```

输完程序后，键入“SYSTEM”命令（在 BS. 2A 版本下）或“BGTOOL”（在 V3 版本下）命令进入键控绘图状态，按图 7—27

规则绘图。

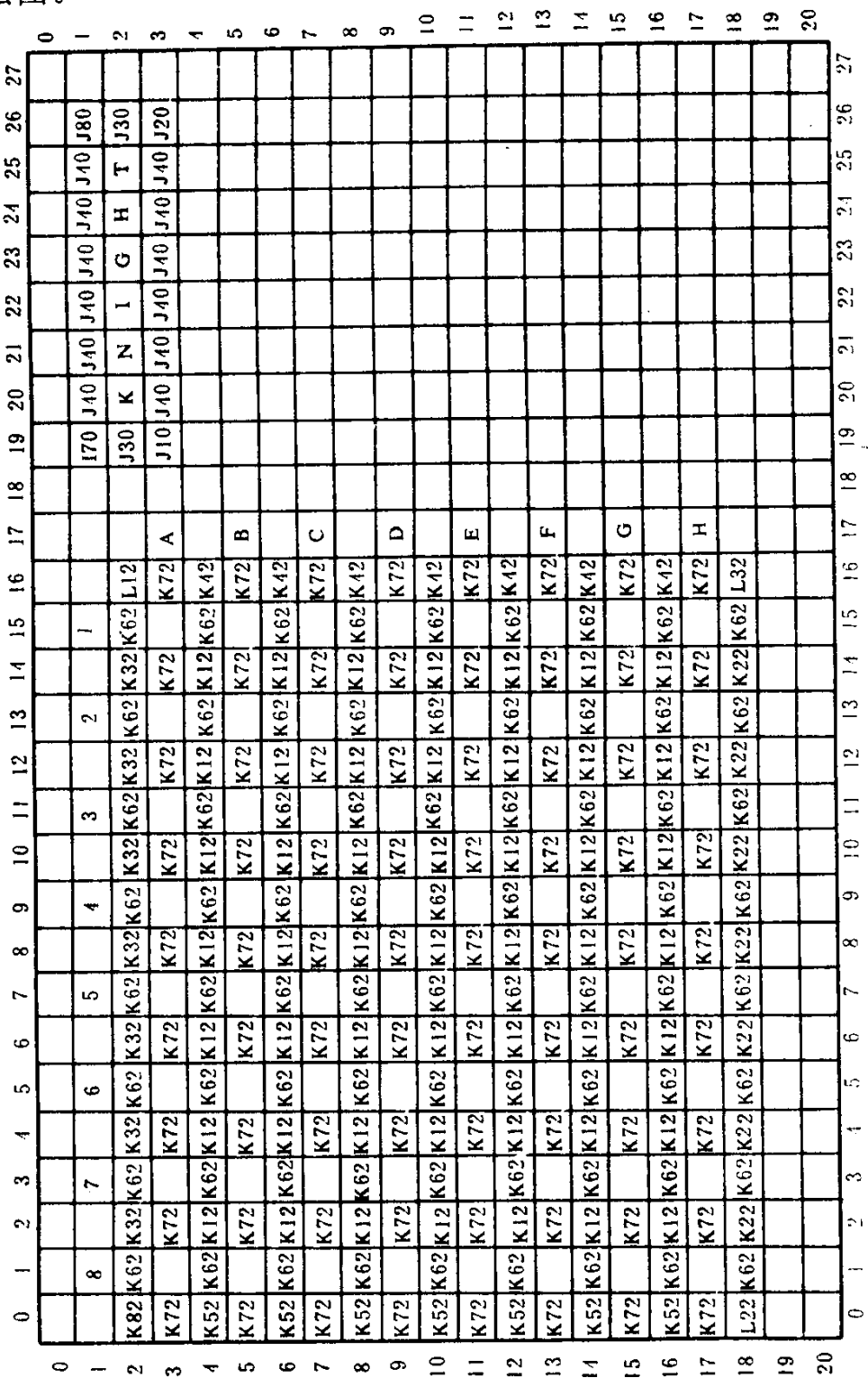


图 7—27 《跳马》游戏画面绘图规则

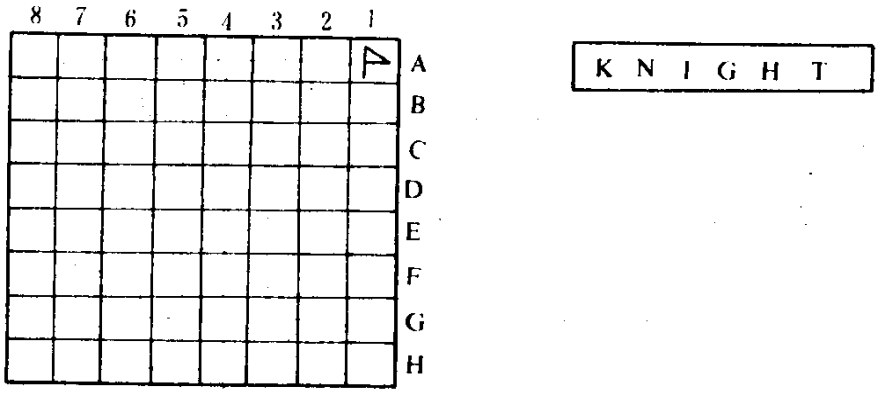


图 7—28 《跳马》游戏画面

绘图完毕，返回 F BASIC 程序状态。运行程序，显示图 7—28 的游戏画面。游戏者双方可使用操纵器的方向键上、下、左、右控制小旗移动（小旗表示下棋和走棋的位置），每按一次，在该方向上移动一格，以此来选定棋子的落位。落位选定后，按 A 键就在选定的位置上出现棋子。然后小旗又出现在右上角，由游戏对方来选择第一步棋子的落位。双方选定棋子落位后，即开始走棋。I# 操纵器控制红棋（RED）Ⅱ# 操作器控制蓝棋（BLUE）。这时，游戏双方只能使用操纵器的上、下键控制小旗移动。按向上的键，小旗以顺时针方向顺序指示可走棋的位置；按向下的键则以逆时针方向指示。选定走棋位置后按“A”键，棋子就会移到小旗指示的位置，并在棋子原来所在位置打印“＊”标记。然后小旗会移到对方棋子周围的某一个位置，由对手使用另一个操纵器控制走棋。

在用操纵器控制小旗移动时，当走到有“＊”的位置，小旗会自动跳过。直到有一方无路可走时（即可走的位置都打印了“＊”号），（如红棋无路可走）则游戏结束，显示“BLUE WIN!”（蓝棋胜）。

程序说明：

（1）20～70 行定义棋子和小旗以及它们的颜色。棋子有红、蓝两色。

（2）90 行定义 HX（1）、HY（1）两个数组用来装红、蓝两棋子的

位置坐标；X(7)、Y(7)两数组装每个棋子可走的八个位置的相对偏移量；B(7,7)数组用来记忆某一位置是否走过，B 数组共有 64 个下标变量，每一个下标变量对应棋盘中的一个空格，若 B(X1，Y1)＝1 则表示(X1，Y1)位置已有棋子走过，禁止再走，若 B(X2，Y2)＝0 则表示该位置没有棋子走过，可以走。

(3)100、110 两行语句对棋子能走的八个位置的相对坐标(相对于棋子，取 Y 轴向下、X 轴向右)赋值。图 7—26 中的八个相对座标依次为：①(－1，－2)、②(－2，－1)③(－2，1)、④(－1，2)、⑤(1，2)、⑥(2，1)、⑦(2，－1)、⑧(1，－2)。

(4)190、200 两行通过调用 250、440 开头的子程序完成游戏者双方第一步棋子的落位。190 行控制第一位游戏者的棋子落位，200 行控制另一位。

(5)210～240 行使 C＝0 或 C＝1，控制游戏者轮流走棋。其中 230 行设置结束转移出口，F＝－1 标志着八个方向的位置都已走过，即无路可走了。转结束处理程序。

(6)250～330 行为第一步棋子落位时，小旗移动的控制子程序，通过该子程序计算小旗上、下、左、右移动的位置坐标，通过调用定位子程序显示小旗。该子程序在双方第一步棋子落位后便不再调用。

(7)340～430 行为游戏中棋子可行的八个位置的确定及判断某位置是否走过的子程序。其中 360～380 行根据方向键值控制选择的顺、逆时针方向；390 行控制按跳马的规则，自动选择①～⑧的位置。即该语句保证了小旗的位置(X，Y)只能在[HX(C)、HY(C)]周围的①～⑧的八个位置内选择；400 行使用 F 变量作位置计数器，若八个位置都已走过则令 F＝－1，由 230 行转结束处理；420 行控制小旗跳过已走过的位置，在未走过的位置显示。

(8)440～550 行为定位子程序，当该子程序被调用时总 是有 C＝0 或 C＝1，所以 440 行的功能就是使小旗根据 X、Y 的取值显

示在屏幕上。450、460行的作用是读操纵器的键值。若按了A键则令B(X,Y)=1,记下此位置已有棋子走过,然后通过500、510语句让棋子移到新的位置,并通过520、530语句在棋子的原位置上设置禁走标志"*"。

(9)560～600为结束处理程序。它将根据C的取值打印"BLUE WIN!"(蓝胜)或"RED WIN!"(红胜)的评判。

7.5.3 结合特技

结合特技包括:字符显示与音响的合成;背景图形与程序的合成;卡通图案与背景图案的结合;动态背景与卡通图案的合成。

1.字符显示与音响的合成

诸如《人间兵器》、《恶魔城传说》等电视游戏,在打印游戏说明或字符提示时,伴随着滴滴的音响,把字符逐个打印在屏幕上,使枯燥的字符显示变得生动活泼。现介绍实现这一合成的方法。

例如:我们欲在屏幕上开一个横座标为6～22、纵座标为4～15的窗口,打印如下一段文字:

"Wind and rain escorted Spring's departure,

Flying snow welcomes Spring's return.

On the ice—clad rock rising high and sheer

A flower blooms sweet and fair."

(毛泽东诗词:风雨送春归,飞雪迎春到,已是悬崖百丈冰,犹有花枝俏。)

实现程序见No.7—30

```
5 REM "No.7—30"
10 CLS
20 PLAY "T1M1V5Y2"
30 DIM A$(10)
40 FOR I=0 TO 10
50 IF I MOD 3=0 THEN A=A+1
```

```
60 LOCATE 6,4+I+A:READ A$(I)
70 FOR J=0 TO LEN(A$(I))
80 C$=MID$(A$(I),J+1,1)
90 PRINT C$;:PLAY "O3G2"
100 NEXT
110 PAUSE 50:NEXT
120 PLAY "O3E2DCDE":LOCATE 0,22:END
130 DATA "Wind and rain es","－corted Spring's","departure,",
    "Flying snow welc","--omes Spring's","return"
140 DATA "On the ice－clad","rock rising high","and sheer","A
    flower blooms","sweet and fair."
```

程序中50行的作用是，在字符打印时，每打印完一句，空出一行，以便于阅读。

2. 背景图形与程序的合成

在某些游戏软件中，往往把背景图案直接写在语句之间。这样做的好处是：

- 程序直观、便于阅读理解；
- 可使绘图程序得到极大地简化；
- 可提高背景图形的显示、打印速度；
- 便于背景图形整块移动；
- 在游戏的换场中，可把每一场的图形装入一个数组，换场时通过循环语句、再配合“SCREEN”指令，可迅速完成换场。

但是，众所周知，背景图形只能通过使用“PRINT CHR$(XXX)”语句打印在屏幕上，怎么能写入程序中呢？

实现的方法如下：

(1)用“BGTOOL”或“SYSTEM”命令进入BG GRAPHIC绘画状态，进行键控绘图。

(2)绘图完毕，先按“ESC”键、再按“STOP”键，返回BASIC状态。

(3)用“VIEW”指令调出绘制的画面。

(4)把画面中每一行的图形作为一个字符串，装入字符串数组中。一般每一版图形用一个串数组，如A$(20)、B$(20)等，每一行前面加入一个行号。

(5)把0～20行图形均装入程序中后，当需调用某版画面时，只要使用

```
FOR I=0 TO 20:RPINT A$(I):NEXT
```

就可立即调出画面。

注意：

·当用VIEW命令调图形时，可能出现指令执行完毕提示符“OK”破坏画面的情况。这时可使用LOCATE X,Y:VIEW命令把“OK”移到其它位置。

·对某行开始就有图形的情况，可用“Ins”键移出打印行号和变量名的空位。

·每行图形写入变量时的开始位置要固定，否则将破坏画面的完整、统一。

·使用这种方法绘图时，不要用背景图形库中前四组图形，否则将不能显示。

3. 卡通与背景的结合

在《敲冰块》、《冒险岛》、《超级玛丽》等游戏中的主人翁，可在操纵器的控制下，跳上台阶、越过深涧、登上楼梯、跃下悬崖，在背景画面中蹦跳随意、运动自如。这种效果就是卡通运动与背景画面有机结合而产生的。也就是说，在背景画面中，卡通的运动已不再是独立的，它要受到画面的各种限制。卡通在每一个动作执行以前，要首先检查其立足点是什么背景，再检查按操纵器指示的前进方向上又是什么背景，若指示的动作合理则执行，若不合理则拒绝执行。如若在卡通的面前是一堵墙，操纵器给出向前走的指令则拒绝执行；若卡通目前的立足点是平地，其左、右也是平地，则可以执

行操纵器左、右走的指令；若在向上的方向上没有梯子或攀援物(如绳、索等)则拒绝执行向上的指令；若立足点是楼梯，其上、下方向也是楼梯则可以执行上、下动作指令；若其左、右方向都是空，令其左、右动作时则只能作跃下的动作，直到降落在有形的物体上等等。总之在背景画面中的卡通运动已不再是机械地执行操纵器的任何指令，而是有一定的人工智能判断、有选择地执行。

下面介绍卡通与背景万能结合程序 No. 7—31(工具程序)。使用该程序，只要配备适当的背景画面，变更程序中的个别数据，就可操纵卡通在背景中任意运动。

```
5 REM "No. 7—31"
10 CGSET 1,0
15 ON ERROR GOTO 265
20 CLEAR
25 PALETB 3,13,48,54,22:SPRITE ON
30 DEF SPRITE 0,(0,1,0,1,0)=CHR$(25)+CHR$(24)+CHR
   $(27)+CHR$(26)
40 DEF SPRITE 1,(0,1,0,1,0)=CHR$(13)+CHR$(12)+CHR
   $(15)+CHR$(14)
50 DEF SPRITE 2,(0,1,0,0,0)=CHR$(12)+CHR$(13)+CHR
   $(14)+CHR$(15)
60 DEF SPRITE 3,(0,1,0,0,0)=CHR$(20)+CHR$(21)+CHR
   $(22)+CHR$(23)
70 R$=CHR$(215):H$=CHR$(210):U$=CHR$(192):DIM
   Q(3),D(3):X=116:Y=127:POSITION 7,X,Y:VIEW:K=0:V=
   0:GOTO 170
80 S1=S:S=STICK(0):IF S>2 THEN S=3-(S>7)
90 X=XPOS(7):Y=YPOS(7):XL=X/8-1:YL=(Y-2)/8
100 ON S GOTO 175,175,140,155
105 V=0:K=0:CUT 7
110 IF SCR$(XL,YL)=" " THEN IF SCR$(XL,YL-1)<>H$
```

```
    THEN GOSUB 260:GOTO 210
115 IF STRIG(0)<>8 THEN 80
120 IF SCR$(XL-1,YL)=U$ THEN 215
125 IF SCR$(XL,YL)=U$ THEN 215
130 IF SCR$(XL+1,YL)=U$ THEN 215
135 GOTO 80
140 V=5:G$=SCR$(XL,YL)
145 IF G$<>H$ THEN IF G$<>U$ THEN S=0:V=0:GOTO
    210
150 GOTO 160
155 V=1:G$=SCR$(XL,YL-1)
160 IF G$<>H$ THEN S=0:GOTO 105
165 IF S1=0 OR K>0 THEN GOSUB 260:K=0
170 DEF MOVE(7)=SPRITE(0,V,1,4):MOVE 7:SPRITE S2:
    PAUSE 5:GOTO 80
175 V=4*S-1:K=1-(K+(K>3))*(S=S1)
180 G$=SCR$(XL+1-2*(S=2),XL):S$=SCR$(XL,YL)
185 IF G$<>U$ AND G$<>H$ THEN X=X-6*(S$<>"
    ")*((S=1)-(S=2)):GOTO 210
190 IF S$="" THEN S=S1:GOTO 210
195 IF STRIG(0)=8 THEN IF G$=U$ OR S$=U$ THEN 215
200 IF K<4 THEN DEF MOVE(7)=SPRITE(0,V,4-K,100):
    MOVE 7:SPRITE S2
205 GOTO 80
208 SPRITE S:SPRITE S1:SPRITE S2:ERA 7
210 G=2:Y=Y+3:GOTO 220
215 G=-9:X=X+(2*K+6)*((S=2)-(S=1)):Y=Y-12
220 SPRITE S,X,Y:ERA 7:SPRITE 3:POSITION 7,X,Y:S2=S:K
    =K*((S=2)-(S=1))
225 G=G+3-3*(G>6):X=X-K:Y=Y+G:SPRITE S,X,Y:PO-
    SITION 7,X,Y
```

```
230 XL=X/8-1:YL=(Y-2)/8:G$=SCR$(XL,YL):S$=SCR
    $(XL,YL-1)
235 IF G$=U$ THEN GOSUB 255:K=1:GOTO 170
240 IF S$=H$ THEN GOSUB 255:SPRITE 3,X,Y:SPRITE S2:S2
    =3:GOTO 80
245 IF G$=H$ THEN GOSUB 255:V=0:GOTO 170
250 GOTO 225
255 GOSUB 260:K=0:SPRITE S,X,Y:RETURN
260 CUT 7:X=XL*8+12:Y=YL*8+7:POSITION 7,X,Y:RE-
    TURN
265 IF ERL=180 THEN XL=-27*(XL<1):GOSUB 260:MOVE
    7:RESUME 200
270 IF ERL=110 THEN XL=-27*(XL<0):GOSUB 260:S=S1:
    RESUME 175
275 IF YL>23 THEN RESUME 208
280 IF XL>27 THEN XL=0:GOSUB 260:RESUME
285 IF XL<0 THEN XL=27:GOSUB 260:RESUME
300 IF XL+1>27 THEN XL=0:GOSUB 260:RESUME
305 IF XL-1<0 THEN XL=27:GOSUB 260:RESUME
310 IF X<12 THEN X=228:RESUME
315 IF X>228 THEN X=12:RESUME
320 END
```

程序说明：

(1)10～70 行为初始化赋值和定义显示卡通程序。其中 30 行定义玛丽跌下，40 行定义玛丽面向右跳，50 行定义玛丽面向左跳，60 行定义玛丽背面，70 行选 H8 号背景图案为宝贝、选 H3 号为楼梯、选 F1 号为地面。读者欲改用其它背景图案绘图，只要把相应图形的代码赋值给 R$、H$、U$即可。

(2)80～110 行为读方向键程序。

(3)115～165 行语句的作用是对按 A 键(跳跃键)后的有关处

理和按上、下键的处理。

(4)175～205 行为按左、右键的处理程序，同时考虑按 A 键后的处理。

(5)210～250 行为上、下移动后和跳跃后的处理程序。其中 210 行令其下移；215 行令起跳跃，若同时按左、右键则在 X 方向产生位移；225 行令其下移过一次后再下移 5 点，或者跳跃后降落时每次降 3 点。

(6)265～315 行为出错处理程序。

(7)170 行确定 7# 卡通上、下运动方向和启动卡通运动。

(8)225、260 行为确定运动卡通起始座标子程序。

(9)200 行确定 7# 卡通左右运动方向和启动卡通运动。

为压缩程序量，该程序采用顺序结构编写，程序的执行过程如下：

程序运行后，首先检测 I# 操纵器的方向键，若按左、右移动键则转 175 行，根据键值确定运动方向，并对横向运动标志 K 赋值。然后检查 7# 动作卡通当前位置的情况和运动方向前一位的情况，若均为“地”则执行 200 行，启动 7# 卡通运动；若有一处不是“地”，则令 7#SP 在纵向下降(即跃下)。下降一次后，测试落脚点是否为“地”，若为“地”则转 170 行向指示方向运动；若仍不是“地”，则继续下降，直到落脚点是“地”为止。

若游戏者按向上键则转 155 行，置运动方向 V＝1。检查当前位置的上一坐标是否有梯子，若有梯子则令其上移 4 点；若没有则不执行上动指令，保持原地不动。

若游戏者按向下键则转 140 行，置运动方向 V＝5。检查当前位置的下一坐标是否有梯子，若有梯子则下移 4 点；若没有梯子而是“地”则原地不动；若既无梯子又无“地”(即为空位)则令其下降。

若游戏者未按方向键，仅按 A 键，这时若当前位置和其前、后都是“地”则转 215 行令 SP 跳起，而后下落。

若游戏者在按左(右)键的同时又按 A 键,则通过 215 行在向上跳起的同时在 X 方向产生位移。

程序中引入变量 G 的目的是使 7# 跳起后下降的动作或跃下的动作分段进行,逐渐下降到地,防止下降速度太快而失去真实感。

注意:

(1)程序 No. 7—31 只有配上背景画面后才能使用。读者欲演示该程序,可使用 BGTOOL 命令调 BG GRAPHIC,使用背景图形库中 F1 号图形画地面,H3 号图形画楼梯,H8 号图形为宝贝设计背景画面。画面设计好后,返回 BASIC,运行程序即可演示。

(2)该程序只适用 V3 版本,若用于 BS. 2A 版本需进行改造。

4. 动态背景与卡通的结合

本节第一部分介绍了动态背景,现在介绍动态背景与卡通图案的结合技巧。运行程序 No. 7—32,屏幕左侧显示一条太空通道,一艘太空船在太空通道上端发出滴滴声向下飞速降落,各种形状的太空陨石从飞船两边飞过。用 I# 操纵器控制飞船左、右移动,躲避陨石的撞击,计数器记录着飞船降落的距离。若飞船降落 1 千米没有与陨石相撞则表示降落成功,打印贺词、停机。若飞船在降落的过程中发生碰撞则计数器从 0 开始。

本程序短小精悍,自成游戏片断;情节激烈有趣,富有冒险精神。若再配以开始、结束画面和难度选择将构成一个完整的游戏。

注:所谓"太空通道"是笔者杜撰的。它的定义如下:一艘太空船欲降落到某个星球上进行探险考察,但这个星球的外层周围布满了厚厚的、密密的太空陨石(这些陨石既不降落、也不离去,悬浮在星球的上空。),飞船承受不了这么多陨石的撞击,所以无法降落。通过绕星球飞行,发现了从空中通往星球的一条圆筒形的道路(或称通道),通道中的陨石较稀疏,飞船可以躲过陨石而降落。因而取名曰"太空通道"。

例 7.6　太空探险

```
5 REM "No. 7—32. OUTER SPACE EXPLOR ATION"
10 CLS:SPRITE ON:CGSET 0,1
20 PLAY "V15Y2T2:V15Y3T2"
30 DEF MOVE(7)=SPRITE(7,1,1,1,0,2):POSITION 7,80.40:
   MOVE 7
40 DIM A$(15):A$(15)=CHR$(199):B$=CHR$(244)
50 FOR I=0 TO 14:A$(I)=CHR$(207+I):NEXT
60 LOCATE 0,21:PRINT A$(RND(16));B$
70 IF D=2 THEN LOCATE RND(15)+2,22:PRINT A$(RND
   (16)):D=0
80 LOCATE 17,23:PRINT B$,A$(RND(16))
90 T=T+1:IF T MOD 10=0 THEN LOCATE 20,20:PRINT "SC="
   T"    "
100 IF T=1000 THEN PLAY "O3E0DCDE":ERA T:CLS:LOCATE
    8,10:PRINT"VERY GOOD!!!":LOCATE 0,20:END
110 D=D+1:S=0
120 K=STICK(0):IF(K=1)+(K=2)=0 THEN 160
130 IF K=1 THEN S=3
140 IF K=2 THEN S=7
150 IF F=1 THEN F=0:GOTO 190
160 PX=(XPOS(7)-16)/8
170 L$=SCR$(PX-1,2):R$=SCR$(PX+1,2):F$=SCR
    $(PX,2)
180 IF L$<>" "OR R$<>" "OR F$<>" "THEN F=1:GOTO
    210
185 IF S=0 GOTO 195
190 DEF MOVE(7)=SPRITE(7,S,1,4,0,2):POSITION 7,PX*8+
    16,1:MOVE 7
195 PLAY "O2#G0"
200 GOTO 60
```

```
210 PLAY "O1E1C:O0#E1C"
220 T = 0: LOCATE 20, 22: PRINT "AGAIN!": LOCATE 20, 20:
    PRINT "SC= 0":GOTO 60
230 END
```

程序说明：

(1)30 行定义太空船。40、50 行选定太空陨石形状装入数组 A $(15)，选定通道边界装入 B$。

(2)60 行画通道左边界。70 行打印通道中的陨石(障碍物)，引入变量 D 控制障碍物的疏密，D 值越大，障碍物越少。80 行画通道的右边界。

(3)90、100 行完成记数、分数显示和降落成功后的简单结束处理。

(4)120～140 行读操纵器的方向键值。

(5)150～180 行判断太空船是否与通道边界或障碍物相撞。若相撞则发声，记数从 0 开始；若没有相撞则执行游戏者的指令，飞船左、右运动，并发出滴滴声。

(6)185～200 行使飞船运动。

(7)210～220 行使记数从 0 开始。

运行程序。

现在你就是一名飞船驾驶员，愿你能沉着冷静地驾驶飞船、机智灵活地躲避陨石、安全平安地降落着陆。该程序可在所有 F BASIC 版本的系统中运行。

程序 No. 7—32 中的动态背景只能向单一方向运动(向上)，下面再介绍一个动态背景可以双向运动的游戏程序。

例 7.7 赛车游戏——《66 号公路》

《66 号公路》是由 1 人玩的赛车游戏，游戏中，游戏者必须在规定时间内跑完全程。用 I# 操纵器的方向键控制赛车左、右偏移，躲避跑道上的其它干扰车辆。若与干扰车辆相撞，赛车会停止运

行，若发生四次碰撞则游戏结束。操纵器上的A键为加速键，B键为刹车键。加速时，赛车会超过干扰车辆；刹车时，干扰车辆会超过赛车。游戏难度分许多等级，等级越高，难度越大，即赛车的最大车速越大，在规定时间内所跑完的路程越长。游戏开始，难度为最低级，若在规定时间内跑院 了全程，则游戏难度会自动升一级，路程会自动增加，游戏继续进行；若在规定时间内未跑完全程，则难度等级不变，继续在这一难度级别下进行赛车。

《66号公路》赛车游戏程序清单如下：

```
5 REM "No.7—33,AUTOMOBILE RACE"
10 VIEW:CGSET 1,2:PALETS 0,13,22,39,2:SPRITE ON:PLAY
   "T1O3"
20 X1=150:X2=170:X3=190:X6=136:X7=208:C0$=CHR
   $(136)+CHR$(137)+CHR$(138)+CHR$(139):C1$=CHR
   $(140)+CHR$(141)+CHR$(142)+CHR$(143):B$=CHR
   $(214)
30 FOR I=4 TO 7:DEF SPRITE I,(0,0,0,0,0)=B$:NEXT
40 F=2:SC=0:CA=3:L=0:X=208:Y0=150:V=0
50 L=L+1:M=L*3+15:MV=M+33:D=(M+27)*400
60 C=0
70 S=STICK(0):T=STRIG(0):V=V-1
80 IF S=1 THEN X=X+5:IF X>203 THEN X=X-5
90 IF S=2 THEN X=X-5:IF X<133 THEN X=X+5
100 IF T=8 THEN V=V+4:IF V>MV THEN V=MV
110 IF T=4 THEN V=V-4
120 IF V<1 THEN V=0
130 IF X<143 OR X>193 THEN 310
140 F=1
150 Y=Y0-V/2:DEF SPRITE 0,(0,1,0,0,0)=C0$:SPRITE 0,X,
    Y:SWAP C0$,C1$
160 IF F1=-1 AND F2=-1 AND F3=-1 THEN SC=SC+1:F1=
```

```
0:F2=0:F3=0
170 FOR I=1 TO 3:DEF SPRITE I,(1,1,0,0,0)=C0$:NEXT
180 YY=(40-Y5)*(Y5>40):SPRITE 1,X1,Y1:SPRITE 2,X2,
    Y2:SPRITE 3,X3,Y3:SPRITE 4,X6,Y5:SPRITE 5,X7,Y5:
    SPRITE 6,X6,YY:SPRITE 7,X7,YY
190 IF ABS(Y1-Y)<12 THEN F1=V>V1:IF ABS(X1-X)<10
    THEN 340
200 IF ABS(Y2-Y)<12 THEN F2=V>V2:IF ABS(X2-X)<10
    THEN 340
210 IF ABS(Y3-Y)<15 THEN F3=V>V3:IF ABS(X3-X)<10
    THEN 340
220 Y1=Y1+V-V1:Y2=Y2+V-V2:Y3=Y3+V- V3:Y5=Y5+
    V:XX=XX+2:X1=ABS(XX)+143:IF XX>48 THEN XX=-
    51
230 IF Y1>250 THEN V1=RND(M)+30:Y1=0
240 IF Y2>250 THEN Y2=0:X2=RND(58)+138:V2=RND(M)+
    30
250 IF Y3>250 THEN Y3=0:X3=RND(58)+138:V3=RND(M)+
    30
260 Y5=-Y5*(Y5<250):Y1=Y1-(Y1<0)*255:Y2=Y2-(Y2
    <0)*255:Y3=Y3-(Y3<0)*255
270 D=D-V:IF C MOD 100=0 THEN GOSUB 380
280 IF D<0 THEN GOSUB 380:PRINT "GOOD!":PLAY "CGDAB":
    SC=SC+(500-C)/10:GOTO 50
290 C=C+1:IF C<401 THEN 70
300 PRINT "TIME IS OVER!":PLAY "BAGFEDC":F=3:L=L-1:
    GOTO 340
310 V=V-4:IF F=2 THEN V=V+4:IF V>MV-15 THEN V=
    MV-15
320 IF V<0 THEN V=0:F=2
330 GOTO 150
```

```
340 PLAY "BAEDC":CA=CA-1:IF CA>-1 THEN X=208:V=0:
    GOSUB 380:ON F GOTO 70,70,30
350 PRINT "GAME OVER!":PRINT "TRY AGAIN"
360 IF STRIG(0)<>1 THEN 360
370 GOSUB 380:RAUSE 500:GOTO 10
380 IF SC>HI THEN HI=SC
390 LOCATE 0,0:PRINT "HIGH SCORE"HI:PRINT "LEVEL"L:
    PRINT "SCORE"SC: PRINT "CARS"CA: PRINT: PRINT:
    PRINT "LEFT"D/4"M":PRINT:PRINT "LEFT"400-C"SEC":
    PRINT:RETURN
```

键入程序后，再键入“SYSTEM”或“BGTOOL”进入键控绘图状态，对照图 7—29 绘图。

绘图后返回编程状态，运行程序，将显示图 7—30 的游戏画面，画面中的英文提示如下：

HIGH SCORE	最高游戏得分
LEVEL	游戏难度等级
SCORE	本次游戏得分
CARS	汽车数(允许碰撞次数)
LEFT	剩余数
SEC	秒
M	米

提示的意思是：

最高游戏得分	180
游戏难度等级	3
本次游戏得分	35
还允许碰车次数	2
剩余路程	2180 米
剩余时间	80 秒

图 7—29 《66 号公路》游戏画面绘图规则

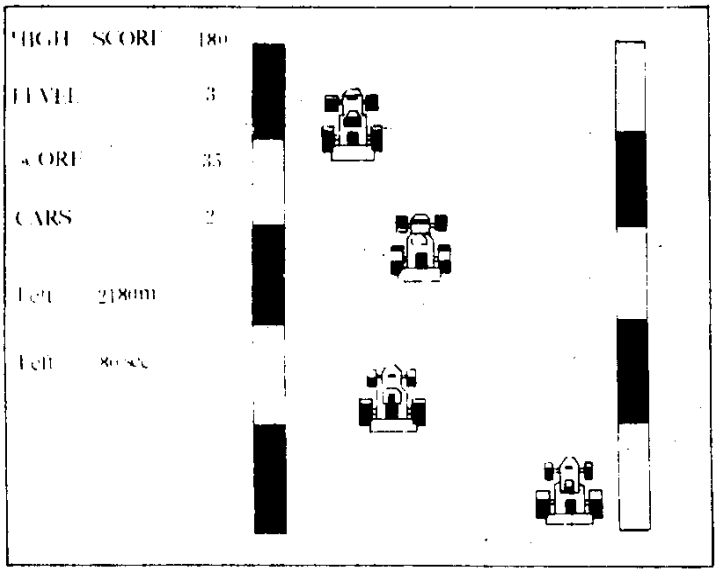


图 7—30 《66 号公路》游戏画面

这时，三辆干扰汽车随机地运行在跑道上，赛车停在跑道的右下边。按 A 键加速，操纵方向键驶入跑道，继续加速，躲避超过干扰车辆，同时跑道两边的标志物迅速后移，提示栏不时地改变所剩余的路程和时间。若按 B 键则车速降低，跑道标志物逐渐不动，干扰车辆纷纷超过赛车。本游戏就是通过跑道标志物下移速度的快慢、干扰车辆上行或下行来产生赛车加速、行驶、刹车的动感。

程序分析如下：

本程序共定义了 8 个卡通：0# 为赛车；1#、2#、3# 为干扰车；4#、5#、6#、7# 为跑道边上的标志物。各卡通的位置座标分别为(X,Y)、(X1,Y1)、(X2,Y2)、(X3,Y3)……。第 30 行定义卡通 4#、5#、6#、7#。第 150 行定义赛车，交换字符串变量 C0$ 和 C1$ 是为使赛车的两个姿势轮流出现。170 行定义 1#、2#、3# 干扰车辆。

(1)程序 20 行中字符串变量 C0$、C1$ 是用于 150、170 行定义赛车和干扰车的。B$ 则用于在 30 行定义跑道的标志物。

(2)第 40、50 行给一些变量赋值：L 为难度等级变量、MV 为最高速度、D 为赛车全程、SC 累计得分、CA 用来记录撞车的次

数、X 是赛车的 X 座标、Y0 是赛车的初始座标。

(3)第 70～290 行是程序的主体，循环执行，完成以下几个任务：

①70～90 行语句的功能是根据操纵器的方向键值调整赛车的 X 座标。

②100～120 行根据 A、B 键是否按下来调整车速。

③130 行用于判断赛车是否越出跑道。如果赛车驶出了跑道，则从第 130 行跳转到 310 行执行越出跑道的处理。

④150 行根据车速 V 调整赛车的 Y 方向座标，并使赛车出现在调整后的位置上。

⑤160 行完成给赛车加分。加分的条件是赛车超过三个干扰车辆，用 F1、F2、F3 分别等于－1 来作为超过干扰车辆 1#、2#、3# 的标志。这三个标志是由第 190～210 行给出的。

⑥170、180 行与 220～260 行共同作用完成根据车速 V 的大小调整干扰车辆的座标、和跑道标志物的座标，并令它们显示在屏幕上。第 220 语句用来调整 1#、2#、3#、4#、5#SP 的 Y 座档，XX 变量的作用是使 1#SP 在跑道上左、右晃动。230～260 行的作用是使 1#～5#SP 从屏幕底部消失后再从屏幕上边跑道上出现。180 行的作用是使 6#、7#SP 的 Y 座标与 4#、5# 的 Y 座标相差 40 个单位，并使 1#～7#SP 显示。

⑦190～210 行的作用是给出赛车超过标志 F1、F2、F3 三变量的值，并判断赛车是否与干扰车辆相撞。若相撞则转 340 行进行撞车处理。

⑧270～280 行的作用是判断赛车是否跑完全程。其中 270 行使记录剩余路程的变量 D 递减，并且每隔 100 个时间单位调用一次 380 行开头的子程序(提示打印子程序)，显示最高得分、难度等级、游戏者得分、赛车个数、剩余距离、剩余时间等参数；280 行判断赛车是否跑完全程，若在规定时间内跑完全程则显示

"GOOD!",计算得分,显示各个提示参数,跳转 50 行,使难度级别加 1,赛车距离增加,继续进行下一轮赛车。

⑨290 行判断时间是否用完。程序每循环一次,时间变量 C 自动加 1,若 C 小于 400 则返回 70 行继续游戏,若 C 大于 400 则转时间到处理程序。

(4)300 行为时间到处理程序。若 C 大于 400 则显示"TIME IS OVER!"(时间到),然后跳转 340 行检查 4 个赛车是否用完,若未用完,赛车返回起跑位置,继续比赛;否则结束游戏。

(5)310～330 行为赛车越出跑道后的减速处理程序。当赛车越出跑道后由从 130 行跳转本程序,由 310 行令车速 V 减小 4 个单位,若游戏者不使赛车返回跑道,则每次执行 310 行车速递减 4 个单位,直到 V＝0。320 行的作用是辅助 310 使车速最低减小到 0 而不为负值。

(6)340～370 行为撞车、时间到、游戏结束的共用处理程序。撞车、时间到的处理原理已经叙述,不多赘言。仅介绍游戏结束的处理,若四辆赛车全部用完则显示"GAME OVER!"、"TRY AGAIN"(游戏结束,再尝试一次),若按开始键则显示本次游戏的结果,然后重新开始游戏。360 行为读开始键语句。

(7)380～390 行用于在屏幕上显示各项提示的参数。

7.6 游戏程序实例分析

阅读成熟的游戏程序,借鉴前人的经验技巧是提高自己编程技能的重要途径。但是,当我们乍接触一个陌生的游戏程序时,往往一时看不懂,难以确定某些语句、变量的作用,读来读去,理不出头绪。真是读不通、理还乱……。

许多朋友要求指给阅读程序的捷径。我们说,没有捷径,但有一定的规律可循。本节以最有代表性的双打游戏《星际大战》为例,

系统地介绍阅读分析游戏程序的一般方法。

7.6.1 熟悉游戏过程

拿到一个游戏程序，先别急于阅读，首先仔细看一遍游戏简介和使用说明(可能有的游戏没有简介说明，那么只有进行第二步)；然后再玩几遍游戏，从而对该游戏的游戏结构、游戏过程、游戏结局有大体的了解。在使用 V3 版本的 F BASIC 系统中固化有《星际大战》的游戏程序(BS 版本中没有。为使只有 BS 版本的朋友也能练习“游戏程序分析”，在本节的后半部分将分析适用 BS 版本的游戏程序。)清单，按“F4”键就可把程序调入并运行。

程序运行后，屏幕上画出了繁星太空，随后显示出太空卫星杀手和太空卫星，接着询问是双打，还是以电脑作对手单打。按“N”键为单人游戏；按其它任意键则进行双人游戏。游戏中 I# 操纵器控制卫星杀手，II# 控制太空卫星，用方向键可操纵卫星向八个方向运动，按“A”键可发射子弹(镭射，也称激光)，按“B”键可改变卫星的运行速度，游戏的目的是卫星之间互相击毙对方。游戏中还出现飞鱼、太空站、苍蝇、陨石、火球等干扰双方作战。若击毁卫星得 10 分、击中干扰得一分，若与对手或干扰相撞则发生爆炸。

每局每个游戏者有四次游戏机会(四颗卫星)，双打时若一方连毁四颗卫星则失去游戏机会，由电脑代替这一方的游戏者继续陪另一方作战，直到另一方也失去游戏机会则游戏结束。暂停一段时间后，游戏重新开始。若选单人游戏则按“N”键，这时由电脑充当对手进行战斗，游戏者一方仍然只有四颗卫星而电脑一方则有无数颗，直到游戏者失去四颗卫星后，游戏结束。游戏中在屏幕的下方有三行文字提示：第一行提示双方的得分；第二行提示双方所剩卫星的数量和游戏难度等级；最后一行提示游戏的最高得分。

通过玩游戏，我们基本清楚了《星际大战》是一个双打、战争类游戏；游戏目的是保存自己、躲避干扰、击毙敌人；每场只有四次游戏机会；根据提示可随时了解敌、我双方的游戏得分，所剩游戏机

会数量及游戏难度等等。

7.6.2 列出程序清单

《星际大战》是由 F BASIC V7.3 版本中移植过来的游戏程序，原程序中的注释已经删去为便于阅读特加入中文注释。

例 7.8 《星际大战》

```
 5 REM "No.7--34,STAR KILLER"
 10 REM 初始化赋值。开始画面。
 20 T=O=U=C=H=D=I=K=L=M=N=P=S=V=X=Y=MA
    =X0=X1=Y0=Y1=PP=Q
 30 DIM D(1),V(1),F(8),SC(1),S(1)
 40 L=9:MA=2:N=3:V=7:C=7:S(0)=4:S(1)=4
 50 X0=70:Y0=80:X1=110:Y1=60
 60 D(0)=3:V(0)=2:D(1)=7:V(1)=2:F(0)=3:F(1)=3:MA=
    MA+1:U=N:M=1
 70 SPRITE ON:CGSET 1,1:PLAY "T1Y2M1V9O3R1":PALETS 1,
    13,48,38,18
 80 FOR I=15 TO 18:A$=A$+SCR$(I,22):NEXT:H=VAL(A
    $)
 90 GOSUB 1070
 100 VIEW:FOR P=0 TO 200:LOCATE RND(27),RND(23):PRINT
     CHR$(205);:NEXT:P=0
 110 REM 主控程序
 120 I=P+6:IF F(P)THEN ON F(P)GOSUB 670,700,680:GOTO
     150
 130 IF M=P THEN GOSUB 390:GOTO 150
 140 GOSUB 210
 150 IF MOVE(I)=0 THEN ERA I
 160 GOSUB 530
 170 GOSUB 800
 180 ON F(8)+1 GOSUB 940,1040
```

```
190 P=1-P:GOTO 120
200 REM 使用操纵器
210 S=STICK(P):T=STRIG(P)
220 IF T<>4 THEN F(I)=1
230 IF S+T=0 THEN RETURN
240 IF S=0 THEN 350
250 S=-1*(S=8)-3*(S=1)-5*(S=4)-7*(S=2)-2*(S=
    9)-4*(S=5)-6*(S=6)-8*(S=10)
260 Q=D(P)
270 IF S=Q THEN 350
280 IF Q>4 THEN 310
290 IF (S>Q AND S<=Q+4)THEN 320
300 GOTO 330
310 IF (S>=Q-4 AND S<Q)THEN 330
320 Q=Q+1:GOTO 340
330 Q=Q-1
340 PP=-1*(Q>8)-8*(Q<1):Q=-Q*(PP=0)+PP:D(P)=
    Q
350 IF T AND 4 THEN IF F(I) THEN V(P)=V-V(P):F(I)=0
360 DEF MOVE(P)=SPRITE(P+8,D(P),V(P),255,1):MOVE P
370 IF T>7 THEN DEF MOVE(I)=SPRITE(12,D(P),1,80,0,P):
    POSITION I,(XPOS(P)+4)MOD 256,(YPOS(P)+4)MOD 256:
    MOVE I
380 RETURN
390 REM 电脑指挥
400 K=K-1:IF K>0 THEN RETURN
410 T=0:D=0:X=(XPOS(1-P)-XPOS(P))/13:Y=(YPOS(1-
    P)-YPOS(P))/13
420 IF X>0 THEN D=3
430 IF X<0 THEN D=7
440 IF Y>0 THEN D=D+5
```

```
450 IF Y<0 THEN D=D+1:IF X<0 THEN D=16
460 IF X AND Y THEN D=D/2
470 S=2:O=D-D(P):IF(0<O AND O<5)OR O<-3 THEN S=1
480 IF O=0 THEN S=0
490 IF V(1-P)<>V(P)THEN T=4
500 IF MOVE(I)=0 OR RND(10)=0 THEN T=T+8
510 IF X*X+Y*Y>200 THEN RETURN
520 K=RND(L):GOTO 220
530 REM 碰撞测试
540 D=CRASH(0):IF D>0 AND D<6 THEN O=0:GOSUB 590
550 D=CRASH(1):IF D>1 AND D<6 THEN O=1:GOSUB 590
560 D=CRASH(6):IF D>0 AND D<6 THEN O=6:GOSUB 610
570 D=CRASH(7):IF D=0 OR (D>1 AND D<6)THEN O=7:GO-
    SUB 610
580 RETURN
590 X=ABS(XPOS(O)-XPOS(D)):Y=ABS(YPOS(O)-YPOS
    (D)):IF X<10 AND Y<10 THEN IF MOVE(O)AND MOVE
    (D)AND F(D)=0 AND F(O)=0 THEN F(O)=1:F(D)=1:IF D
    >1 THEN 650
600 RETURN
610 X=ABS(XPOS(O)-XPOS(D)-4):Y=ABS(YPOS(O)-YPOS
    (D)-4):IF X<8 AND Y<8 THEN IF MOVE(O)AND MOVE
    (D)AND F(D)=0 THEN ERA O:F(D)=1:GOTO 630
620 RETURN
630 IF D<2 THEN SC(O-6)=SC(O-6)+10:RETURN
640 SC(O-6)=SC(O-6)+1+F(8)*4
650 DEF MOVE(D)=SPRITE(10,RND(9),RND(3)+4,10):MOVE
    D:PLAY"O4C":RETURN
660 REM 卫星爆炸处理,文字提示。
670 F(P)=2:DEF MOVE(P)=SPRITE(10,D(P),V(P),70/V(P),
    0):MOVE P:PLAY "O"+HEX$(P+3)+"GG":S(P)=S(P)-
```

```
    1:RETURN
680 IF P=M OR STRIG(P)+STICK(P)THEN F(P)=0:POSITION
    P,X0+P*X1,Y0+P*Y1
690 RETURN
700 IF MOVE(P)THEN RETURN
710 F(P)=3:ERA P
720 IF SC(0)>H AND M<>0 THEN H=SC(0)
730 IF SC(1)>H AND M<>1 THEN H=SC(1)
740 IF L<1 THEN L=5
750 LOCATE 0,20:PRINT "SCORE",SC(0),SC(1),"LEVEL":
    PRINT "SHIPS",S(0),S(1),L;" ":PRINT "HI-SCORE",H.
760 IF S(P)>0 THEN RETURN
770 IF M=P THEN L=L-1:S(P)=4:GOTO 740
780 IF M=-1 THEN M=P:PRINT P"     COM":PLAY "EE":RE-
    TURN
790 LOCATE 10,10:PRINT "GAME OVER":PLAY
    "O1C5CDED6C4C5":PAUSE 200:RUN
800 REM 卫星与陨石相撞的处理。
810 X=XPOS(P)/8-1:Y=YPOS(P)/8-2
820 IF X<0 OR X>27 OR Y<0 OR Y>23 THEN 880
830 IF SCR$(X,Y)<>CHR$(207)THEN 880
840 S=ASC(SCR$(X,Y,1))
850 IF S=P+1 THEN 880
860 IF F(P)=0 THEN F(P)=1
870 REM 激光弹击中陨石
880 IF MOVE(I)=0 THEN RETURN
890 X=(XPOS(I)-12)/8:Y=(YPOS(I)-20)/8
900 IF X<0 OR X>27 OR Y<0 OR Y>23 THEN RETURN
910 IF SCR$(X,Y)<>CHR$(207)THEN RETURN
920 S=ASC(SCR$(X,Y,1)):IF S=P+1 THEN RETURN
930 ERA I:LOCATE X,Y:PRINT " ";:RETURN
```

```
940 REM 定义干扰卡通
950 FOR I=2 TO MA:IF MOVE(I)OR F(I)=2 THEN 980
960 IF F(I)=1 THEN GOSUB 990:GOTO 980
970 DEF MOVE(I)=SPRITE(C,RND(9),RND(3)+C/2,80,0):
    MOVE I
980 NEXT:RETURN
990 REM
1000 F(I)=0:ERA I:POSITION I,120,130:IF U<MA THEN F(I)=
     2
1010 U=U-1:IF U=0 THEN F(2)=0:F(8)=1:RETURN
1020 IF U<0 THEN U=N:F(2)=0:F(3)=0:F(4)=0:F(5)=0:F
     (8)=0:C=C-2:L=L-1:IF C<2 THEN C=7:IF M>-1
     THEN S(1-M)=S(1-M)+1
1030 RETURN
1040 IF F(2) THEN I=2:GOTO 990
1050 IF MOVE(2)=0 THEN DEF MOVE(2)=SPRITE(2,RND(8)+
     1,RND(3)+1,10,0):MOVE 2
1060 RETURN
1070 REM 开始画面与选项
1080 VIEW:RESTORE 1070
1090 FOR P=0 TO 200:LOCATE RND(27),RND(23):PRINT CHR
     $(205):NEXT
1100 LOCATE 8,3:PRINT "STAR KILLER"
1110 FOR P=0 TO 1:LOCATE 10,10+P*8:DEF MOVE(P)=
     SPRITE(P+8,D(P),2,30):POSITION P,X0+P*X1,Y0+P*
     Y1:MOVE P
1120 IF MOVE(P)THEN 1120
1130 READ A$:FOR I=1 TO LEN(A$):PRINT MID$(A$,I,
     1);:PLAY "C":NEXT:NEXT
1140 FOR P=2 TO 3:LOCATE 10,17+P:READ A$:FOR I=1 TO
     LEN(A$):PRINT MID$(A$,I,1):PLAY "C":NEXT:
```

```
    NEXT:LOCATE 1,22:PRINT "2 PLAYER? (Y/N)";:M=-1
1150 A$=INKEY$(0):IF A$="N"THEN M=1
1160 PAUSE 80:RETURN
1170 DATA "PLAYER 1","PLAYER 2","    OR","COMPUTER"
```

注：原程序中 830、910 行中写入了背景图案，为便于印刷，改为 CHR$(207)。

7.6.3 程序分段处理

阅读程序，一般遵循先整体后局部，先易后难的原则。首先从整体上把程序分段；模块结构的程序可根据 GOSUB 语句的行号划分功能模块和子程序；顺序结构的程序一般按 GOTO 或 THEN 语句的行号分段。《星际大战》共有 13 条 GOSUB 语句，行号分别为：210、390、530、590、610、670、680、700、800、940、990、1040、1070，这些行号一般为功能模块或子程序的入口行号。然后再根据 RETURN 语句确定模块或子程序的结束行号，从而可分出 11 个模块。它们分别为：

1. 240～380
2. 390～520
3. 530～580
4. 590～670
5. 680～690
6. 700～790
7. 800～930
8. 940～980
9. 990～1030
10. 1040～1060
11. 1070～1170

不难看出 20～80 行为初始化赋值程序，我们也可把它列为一个模块，称为模块 0，行号为 20～80。这样整个程序共划分 12 个模

块。

7.6.4 写出管理程序

程序分段以后，剔除以上 12 个模块，就可较容易地列出主程序(管理程序)。程序清单如下：

```
10 REM "STAR SKILLER"
20～80
90 GOSUB 1070
100 VIEW:FOR P=0 TO 200:LOCATE RND(27),RND(23):PRINT
    CHR$(205):NEXT:P=0
110 REM "MAIN PROGRAM"
120 I=P+6:IF F(P)THEN ON F(P)GOSUB 670,770,680:GOTO
    150
130 IF M=P THEN GOSUB 390:GOTO 150
140 GOSUB 210
150 IF MOVE(I)=0 THEN ERA I
160 GOSUB 530
170 GOSUB 800
180 ON F(8)+1 GOSUB 940,1040
190 P=1-P:GOTO 120
```

现在清晰多了。从主程序清单可见，整个游戏的过程就是在 120～190 行之间反复运行。

7.6.5 大概确定各模块的功能

写出管理程序后就可分别分析、逐个确定各模块的功能。对于一时较难确定功能的模块，可暂时跳过，待其它模块功能确定后，在详细分析程序时再对其进行专题研究。

下面我们进行这项工作。

1. 模块 0：20～80 语句行。为初始化程序。

2. 模块 1：210～380 行。读操纵器键值，确定卫星的运动方向。启动 8# 卡通——杀手卫星、9# 卡通——太空卫星、12# 卡通——

镭射运动。

3. 模块 2:390～520 行。这一段程序功能不太明显,暂往后放。

4. 模块 3:530～580 行。测试各动作 SP 之间是否发生碰撞或被击中。

5. 模块 4:590～670 行。测试发生碰撞的两卡通之间的距离,若距离较近(小于 10 点)则视为被击中或碰中,启动爆炸卡通 10# 动作。

6. 模块 5:680、690 行为确定 P# 动作 SP(0# 或 1#)的起始座标子程序。

7. 模块 6:700～790 行的功能是打印屏幕提示和结束处理。

8. 模块 7:800～930 行。这段程序较难理解,且对整个程序的运行影响不大,可以删去。删去的理由是屏幕上没有 G8 号图案,因而程序运行中必然从 830 行转 880 行,再返回主程序,而且 810 行计算的 X、Y 值主程序也不使用。因此,可删去主程序中的 170 行。

9. 模块 8:940～980 行。定义 C#SP。从 40 行可知 C 的初值是 7,7#SP 是太空站。其中 960 行还要调用 990 行(即模块 9),而模块 9 又仅为 960 行所调用,所以可把两模块合并。两模块的功能是:定义干扰物。从 1020 行可知 C 分别等于 3、5、7。因此,对应的干扰物分别为太空站、飞鱼、火球。

10. 模块 10:1040～1060 行。定义 2#SP—— 苍蝇并启动其运动。

11. 模块 11:1070～1170 行为标题和选项处理程序。

通过以上的工作,除模块 2、7 的功能暂不明确外,其它模块的功能已大体确定。

7.6.6 追踪主程序的流向

现在我们根据主程序的执行过程,详细分析程序。

分析程序前,最好列出变量表及变量赋值表,以便分析程序时

参考。当每确定一个变量的作用时，则立即在变量表中注明，以利阅读理解。

变量表见表 7—2(由 20、30 程序行得出)。

变量赋值表见表 7—3(由 40～60 行得初值)。在列变量赋值表时要使每个变量后面留出较大的空间，以便追踪程序运行时，记录其值的变化。

表 7—2 《星际大战》变量汇总表

单字符变量	双字符变量	下标变量
C	MA	D(0)
H	PP	D(1)
D	X0	V(0)
I	X1	V(1)
K	Y0	S(0)
L	Y1	S(1)
M		SC(0)
N		SC(1)
O		F(0)
P		F(1)
Q		F(2)
S		F(3)
T		F(4)
U		F(5)
V		F(6)
X		F(7)
Y		F(8)

列好变量表、变量赋值表后，开始分析程序：

1. 第一周期(我们把主程序执行一个循环称为程序运转一个周期。下同。)

(1)由 90 行转 1070 行，可知这一模块的功能是打印标题画面

和游戏对手选项。1090 行语句打印星空，1110、1130 行语句定义 8#、9#SP 为 0#、1# 动作 SP。1140 行打印选项提示。1150 行读键盘输入：若键入“N”则为单人游戏，此时 M＝1；若按其它键则为双人游戏，这时 M＝－1。选项后，返回主程序。

假定我们选双人游戏，则 M＝－1，把“M＝－1”写入变量赋值表。（今后每个变量得到一个新值后，都要及时写入表 7—3，以利分析。在以后的叙述中不再提示，请读者自动写入。）

表 7—3 《星际大战》变量赋值表

C＝7
L＝9
M＝1
N＝3
U＝3
V＝7
MA＝3
X0＝70
Y0＝80
X1＝110
Y1＝60
D(0)＝3
D(1)＝7
V(0)＝2
V(1)＝2
S(0)＝4
S(1)＝4
F(0)＝3
F(1)＝3
F(2)＝
F(3)＝
F(4)＝
F(5)＝
F(6)＝
F(7)＝
F(8)＝

通过 1070～1170 行程序段的分析，可大致确定以下几个变量的作用：

P：0#、1# 动作 SP 的转换变量。P＝0、1。

D(P)：为 0#、1# 动作 SP 的运动方向变量。

M：游戏者单、双打选项变量。M＝1、－1。

(2)由 100 行语句清屏，重新打印星空，而后令 P＝0。游戏过程开始……

(3)120 行，I＝P＋6＝6，F(P)＝F(0)＝3。(该语句行中使用了"IF F(P) THEN……"语句，其作用是"如果 F(P)≠0 则执行 THEN 后面的语句"。"IF F(P)"相当于"IF F(P)<>0"，这种表达方式是电脑约定的简写形式。)由于 F(0)＝3 所以调 680 行。

(4)现在 P＝0、M＝－1、P≠M，假定未按操纵器的任何键则返回 120 行。

(5)由 120 行跳转 150 行。

(6)由 160 行调 530 行。由于游戏刚开始，没有卡通相撞故返回 160 行。

(7)以上我们曾分析过，模块 7 可先跳过，故删去 170 行。

(8)180 行，这时 F(8)＝0，因此调 940 行。

(9)940～980 行，C＝7，MA＝3，定义 7# SP 为 2#、3# 动作 SP，并令其运动。

(10)190 行。P＝1，跳转 120 行，第一个循环结束。

通过第一次循环，启动了 2#、3# 动作 SP(太空站)开始运动，同时确定了以下变量的作用：

D：先为 SP 碰撞记录变量，后为辅测(辅助测试)碰撞 SP 的动作编号。

O：为主测碰撞 SP 的动作编号。

C：为干扰 SP 的编号。

MA：为干扰动作 SP 的数量控制变量。

注：当测试某个 SP 是否碰撞时，如测试 1# 动作 SP 与 2# 动作 SP 相撞，则 O＝1、D＝2，即 1# SP 是主测动作，2# 则是辅测 SP。

2. 第二周期

(1)120 行。P=1、I=7、F(1)=3，调 680 行。假定这时按 Ⅱ 号操纵器方向键。

(2)通过 680 行，F(1)=0，确定 1# 动作 SP 的起始座标为(110,60)。返回 120 行，转 150 行。

(3)通过 160 行调 530 行，没有碰撞，返回。

(4)180 行。这时 F(8)=0 F(8)+1=1，调 940 行。

(5)这时 2#、3# 动作都没有结束，返回。

3. 第三周期

(1)P=0、I=6、F(0)=3，转 680 行。

(2)假定按 I 号操纵器的方向键，则通过 680 行 F(0)=0。确定 0# 动作 SP 的座标为(70,80)。返回 120 行。

(3)转 150 行、调 530 行，没有碰撞，返回。F(8)=1，调 940。这时 2#、3# 动作仍没结束，返回。

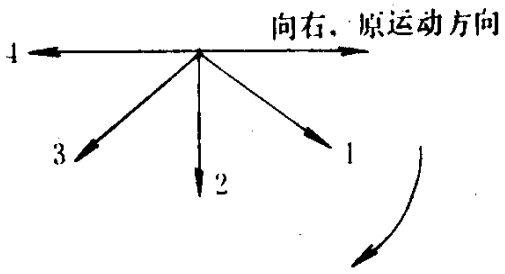
(4)P=1，转 120。第三次循环结束。这时 0#、1# 动作已分别确定了起始座标。

4. 第四周期

(1)P=1、I=7，这时 F(1)=0 则执行 140 行。

(2)由 140 行调 210 行。

(3)210 行。由于我们已经按方向键，假定按右键则 S=1，通过 250 行 S=3。(注意 250 行完成 SP 八个运动方向的计算。这一行语句等价于"IF S=1 THEN V=3"形式的八行语句。请读者记住这一编程技巧。计算运动方向的过程是：当 S=1 时，只有(S=1)=-1，其它各项都为 0，故有 S=-(-1)×3=3。即方向键值为 1 时，运动方向为向右。)260～340 行程序是该模块中独立的一段程序，它的作用是使 0#、1# 动作的转向每次 45°，分步完成。如正在向左运动时，若按键令其下行，则分两步从向左改为向下。若正在向右运动，按键令其向左行时则分四步从右行转向左行。转向过程示意如下：



分步转向的原理留待下一步分析。

运动方向 S 的值通过方向转移变量 Q 逐步转给 D(P)，再通过 360 行定义 1#动作 SP 运动。若在按方向键的同时又按了 A 键(发射键)则定义 7#动作 SP 运动(即发射激光弹)，运动起点在 1#动作 SP 的中心外 4 点的位置，运动方向与 1#SP 相同。返回 150 行。

(4)150 行。若 7#动作结束则令其消失。表现在画面上是激光弹未击中目标，令其消失。

(5)160 行。调 530 检查 0#、1#动作是否与其它动作相撞或除 1#动作 SP 外的各动作是否被击中。假定 0#被击中则由 570 行 D=0、O=7，转 610 行。

(6)610 行。计算 0#、7#动作之间的距离，若在 X 方向大于等于 8，同时在 Y 方向也大于等于 8 则判定未被击中，返回；若有一个方向上的距离小于 8，同时 0#、7#动作 SP 的运动又未结束，且有 F(0)=0 则判定 0#SP 被击中。

(7)判定 0#SP 被击中后，则令 7#SP 消失，令 F(0)=1，转 630 行。

(8)由于 D=0 小于 2，则有 SC(O-6)=SC(7-6)=SC(1)，SC(1)=SC(1)+10=10，作用是给 1#动作 SP(Ⅱ#游戏者)加 10 分。下标变量 SC(0)、SC(1)为记分变量，分别记录 0#、1#动作的得分。返回 570 行。返回 160 行。

(9)由 180 行调 940，检查 2#、3#干扰 SP 的动作是否完成。若动作完成则重新启动运行。可见，每次循环都要“GOSUB 940”对

2#、3#SP 检查一次。这是例行公事，以后不再叙述。

(10)P=0，转 120 行。

5. 第五周期

(1)P=0、I=6、F(0)=1，调 670。

(2)670 行。首先令 F(0)=2，定义 0# 动作 SP 为 10# 卡通(爆炸)，在原运动方向上运行 10～14 点，同时发声，令 S(0)=S(0)-1=4-1=3。由于上一周期判断 0# 动作 SP 被击中，所以演示其被击中的景象。(下标变量 S(0)、S(1)记录 0#、1# 动作 SP 的游戏机会数)返回 120 行，转 150 行。

(3)160 行。调 530，由于 0# 被击中，则返回，P=1-0=1，转 120 行。

按照上述顺序，读者可依次分析，不难读懂各段程序。下面分析本程序中的难点和疑难变量。

7.6.7 攻克程序难点

1. 260～340 行程序段

在第四周期的分析中，曾谈到这段程序的作用是使 0# 或 1# 动作 SP 分步转向。现在讨论其分步转向的过程。

(1)程序开始运行时，P(0)=3，0#SP 向右运动，假定我们按向下键则有 S=5，通过 260 行 Q=3。这时，S>Q、S<=Q+4，符合 290 行的条件，故转 320 行。通过 320 行，Q=Q+1=3+1=4，转 340 行。在 340 行中因 Q=4、1<Q<8，所以 PP=0，Q=-Q*(PP=0)+PP=-4*(-1)+Q=4，则有 D(0)=4。因 D(0)为 0# 动作 SP 的方向变量，所以 0#SP 由向右运动转为向“右下”方向运动。

(2)第二周期管理 1# 动作 SP，故略过。直接分析第三周期。

在第三周期，仍有 S=5，则通过 260 行 Q=D(P)=D(0)=4。Q=4、S=5 仍满足是 290 行条件则转 320 行。通过 320 行 Q=Q+1=5，转 340 行。在 340 行仍有 PP=0，则 Q=5、D(0)=5，从而

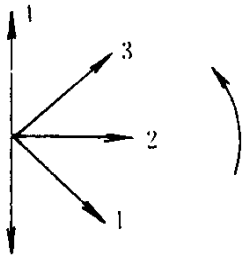
使 0#SP 从向“右下”方向运动改为“向下”运动。共分两步完成了从“右”行到向“下”行的转向。

(3)若在向下运动时，我们又按了上行键，这时 S=1。当程序执行到奇数周期时(0#动作 SP 管理周期)，由 260 行 Q=5，满足 280 行条件转 310 行。现在 S<Q、S=Q−4，满足 310 行条件故转 330 行。通过 330 行，Q=Q−1=4，再通过 340 行，D(0)=4，则 0# 动作由“下”行改向“右下”。

在第五周期：由 260 行 Q=4，通过 300 行转 330 行，Q=Q−1=3，再通过 340 行使 D(0)=3，从而使 0# 动作由向“右下”改为向“右”运动。

在第七周期时，由 260 行 Q=3，通过 300 行转 330 行使 Q=Q−1=2，再通过 340 行使 D(0)=2，从而使 0# 动作由向“右”改向“右上”。

第九周期：由 260 行 Q=2，通过 300 行转 330 行 Q=1，再通过 340 行 D(0)=1，使 0# 动作由向“右上”运动改为向“上”运动。其转向过程如下图：



下(原运动方向)

总之，这段程序的作用就是控制卡通分步转向，以避免方向突变而失去真实感。

2. 390～520 行程序段

在跟踪分析中未讨论该段程序，现分析如下：

(1)该段程序的使用时机

从 130 行可知，这段程序只有在 M=P 时才被调用。我们知

道，变量 P 的取值始终为 0 或 1，M 的取值则由选项时确定取“1”或“－1”(单打时，M＝1；双打时，M＝－1。)，因此，只有在 M＝1(即单打游戏者选电脑作对手游戏时)才满足 M＝P 的条件。由此可知，这段程序只有在单打、1# 动作 SP 管理周期内调用，因此它是 1# 动作的控制管理程序。

(2)分析程序的作用

确定了程序的使用对象后，就可分析程序的作用了。

由 400 行可知，若 K＞0 则返回，后续程序不起作用。因此只有当 K＝1 或 K＝0 时，才执行后续程序。现在假定 K＝0(即未按功能键)，则执行 410 行。由 410 行令 T＝0、D＝0，测试 0#、1# 动作 SP 间的距离，由于卡通的极限座标为(0，0)和(255，255)，我们分别选最小值和最大值，即 0# 座标(0，0)，1# 座标(255，255)，则有 $X_{max}=19$，$Y_{max}=19$。由 420、440、460 三行语句得 D＝3、D＝D＋5＝8、D＝D/2＝4。通过 470 行 S＝2，假定执行该段程序时 1# 动作的运动方向为右，即 D(1)＝3，则有 O＝D－D(1)＝4－3＝1，满足 0＜O＜5 的条件，则令 S＝1。由于 X＊X＋Y＊Y＝19＊19＝361＞200 则返回调用程序。由此得出结论，当 0#、1# 动作相距较远时，该段程序做无效劳动。(即不起作用)

下面再讨论两者距离较小的情况。

设 0# 座标为(80，50)，1# 座标为(180，100)，由 410 行得：T＝0、D＝0、X＝6、Y＝3，由 420、440、460 行得 D＝3、D＝D＋5＝8、D＝D/2＝4，若仍假定 D(1)＝3，则有 O＝1，S＝1。现在 X＊X＋Y＊Y＝6×6＋3×3＝45＜200 则执行 520 行，K＝RND(L)＝RND(9)，跳转 220 行，执行 220～380 行的程序。

另外，在 400～520 程序段中的 490 行的作用是若两卫星的运动方向不一致，则令 T＝4(即改变 1# 的运动速度)。500 行的作用是当发射结束或者 RND(10)＝0 则令 T＝8，或 T＝12(原来 T＝4 时)。(T＝8 表示令 1# 动作继续发射)。

通过以上分析可见，400～520 程序段的作用就是根据 0#、1# 动作的运行情况，自动产生 S、T 的数值，从而代替人控操纵器。通过这段程序可使 1# 动作的运动方向或者向右、或者向左、或者不动，可令 1# 动作不停的改变速度和不定时的发射激光弹，发射间隔以 RND(10)＝0 为依据。程序中引入变量 K 的目的是调整控制 1# 动作的间隔周期数，K 值越大，间隔周期越长。K 值的大小受游戏难度变量 L 的控制，难度越高，L 的值越小，从而 K 值越小，1# 动作的各种动作变化则越频繁，对游戏者的游戏技能要求也越高。

3. 800～930 行程序段

在以上的讨论中，我们抛开了这段程序。在整个程序中也就数这段程序最难理解。实际上，这是编程者故意向使用者卖的关子（出难题），以增加阅读程序的困难（包括删掉了所有的注释语句）。据我分析，原程序中应该还有一段程序，功能是随机打印 G8 号背景图案，作为 0#、1# 动作卡通运动的固定干扰物，犹如太空中的大陨石或小行星。它的干扰效果与火球、苍蝇不同，当卫星碰到固定干扰物时，卫星爆炸而干扰物依然存在。（运动干扰物则与卫星同时爆炸。）但是，固定干扰物却能用激光弹将其击毁，且数量一定，击毁后不再复生。

现在，我们可以补充一段程序以返还其本来面目。补充程序如下：

```
115 FOR I=0 TO 9:LOCATE RND(27),RND(23):PRINT CHR
    $(207):NEXT
```

补充了 115 行程序后，800～930 程序段的作用就不难分析了。

当程序执行到 810 行，首先计算 P#(0、1)动作在 BG 面的座标(X,Y)，若 P# 动作不在 BG 面内则返回调用程序（主程序）；若在 BG 面内，则检查(X,Y)点是否有 G8 号背景图案，若有则令 F(P)＝1（相当于卫星与陨石相撞），然后通过 120 行调 670 行显示爆炸。若(X,Y)点没有 G8 号图案则转 880 行，检查激光弹的动作

是否结束。若激光弹(7#)动作已经结束则返回主程序;若没有结束,则检查激光弹在BG面的座标点上是否有G8号图案,若有则表示陨石被击中,令该陨石消失;否则返回主程序。

4.F数组下标变量作用的分析

F(8)数组共有9个下标变量,其用途分别叙述如下:

·F(0)、F(1):分别记录0#、1#动作SP的运动状态。

·F(2)~F(5):分别记录动态干扰SP的运动状态。

·F(6)、F(7):分别记录0#、1#动作的运行速度调整情况。

·F(8):为定义干扰SP苍蝇的启动控制变量。

各下标变量在程序执行中的变化情况如下:

(1)F(0)、F(1):

以F(0)为例。初始赋值F(0)=3;经680行作用后F(0)=0,表示已确定0#动作卡通的运行起始座标,等待启动;若与其它动作卡通相撞或被击中则F(0)=1(590、610行);0#被击中后,通过670行变为爆炸,这时F(0)=2;爆炸动作运行结束后,通过710行F(0)=3;再启动运行后F(0)=0。

(2)F(2)~F(5)

以F(2)为例。初值F(2)=0;若被激光弹击中或与卫星相撞则F(2)=1(590、610行);若本次定义启动的动态干扰,在虽有被消灭但未全部被消灭时则F(2)=2;若全部被消灭则有F(2)=0。

(3)F(6)、F(7)

以F(6)为例。初值F(6)=0;若按"B"键调整运动速度则F(6)=1(220行);速度调整后F(6)=0(350行)。

(4)F(8)

初值F(8)=0;当动态干扰消失MA个后F(8)=1(1010行);当出现一个苍蝇后F(8)=0(1020行)。

到此为止,《星际大战》游戏程序全部分析完毕,各变量的作用也已基本确定。想必读者的变量表中各变量的作用也已填满。兹

将变量表列出供参考。

·C:动态干扰 SP 编号,取值 3、5、7。

·D:在 390～520 行时用于调整 1# 动作 SP 的方向;在 540～650 行记录被撞方的动作编号。

·H:游戏者的最高得分。

·I:激光弹动作编号,6#、7#。

·K:1# 动作 SP 自动运动时,调整间隔周期的控制变量。

·L:游戏难度记录变量。初值 L=9。难度越高 L 值越小。取值由 9—1。当 L=0 时则重新赋值 L=5。(L 也可称为游戏技能评价变量,9 级最低,1 级最高。)

·M:单、双打记录。单打 M=1;双打 M=-1。

·N:向 U 赋值变量,也可不用,而把 1020 行中的 U 改写为 U=MA+1。

·O:记录发生碰撞或被击中时被测动作的编号。

·P:0#、1# 动作转换变量。

·Q:0#、1# 动作分步转向控制变量。

·S:操纵器的方向键值和方向值。

·T:操纵器的 A、B 键值。按 A 键,T=8,发射;按 B 键,T=4,调整速度。

·U:动态干扰消失记录变量。

·V:0#、1# 动作调速辅助变量。

·X、Y:0#、1#、6#、7# 动作在 BG 面的座标。

·MA:一次出现动态干扰的个数。MA 最大可取 4。增大 MA 可增加游戏难度。

·PP:Q 值调整辅助变量。

·X0、Y0:0# 动作的起始座标。

·X1、Y1:1# 动作的起始座标。

·D(0)、D(1):0#、1# 动作的运动方向。取值分别为 1～8。

· V(0)、V(1)：0#、1#动作的运动速度。取值分别为2、5。

· S(0)、S(1)：双方游戏机会的记录变量。

· SC(0)，SC(1)：双方的得分记录。

· F(0)～F(8)：见以上介绍(略)。

7.6.8 《星际大战》编程技巧小结

通过以上分析，我们可以学习以下几点编程技巧。

1. 清零技巧

20行中令23个变量连等实现各变量清零，比令每个变量等于零节约50个内存单元。

2. 上一场游戏积分的转移技巧

80行巧妙地记录了上一场的最高游戏得分，并将其转移到下一场。

3. 运动方向的计算技巧

250行仅用了一行语句代替了八条"IF—THEN"语句，完成了S值的计算。

4. 卡通运动中分步转向技巧

260～340行使运动卡通的转向分步进行，从而改善了转向突变的生硬感。

5. 卡通的自动运动处理技巧

通过400～520行，用程序代替了手动操纵器，完成了动作SP的各种动作控制。

6. 提示打印定位技巧

750行仅使用了一个定位语句就完成了三行提示、六项内容的打印定位。

7.6.9 《星际大战》游戏的补充建议和说明

1. 增加115行随机打印固定干扰物的程序(这是必需的)。

2. 建议积分超过50分时奖励游戏者一次游戏机会。

3. 建议击毁一个干扰物加10分，以鼓励双方争先攻击固定干

扰物。

4. 建议删去 20 行，因为程序每次运行均对所有变量自动清零，故 20 行似乎多余。

5. 建议卫星运行和发射时配有音响，以增强游戏的效果。

6. 说明：欲增加游戏难度，可调整 MA 的初值和增加固定干扰物的数量。

根据以上建议，可加入以下程序：

```
115 FOR I=0 TO 9:LOCATE RND(27),RND(23):PRINT CHR
    $(207):NEXT
120 I=P+6:IF SC(I-6)>50 THEN SC(I-6)=SC(I-6)-50:S(I-
    6)=S(I-6)+1
125 IF F(P)THEN ON F(P)GOSUB 670,700,680:GOTO 150
365 PLAY "O3E1"
375 PLAY "O1#D0REO0#D3"
915 SC(I-6)=SC(I-6)+10
```

以上程序已合理安排了行号，读者可直接加入。

7.6.10 适用 BS 版本的游戏实例分析

例 7.9 《UFO 驱逐战》

谜一般的不明飞行物——UFO，突然飞临地球的上空，威胁着人类生命的安全。“闪电号”飞船奉命驱逐 UFO，UFO 放出“超智能机器苍蝇”与“闪电号”对战，一场激烈的太空战斗打响了。

《UFO 驱逐战》程序清单如下：

```
5 REM "No. 7—35. UFO"
10 VIEW:PLAY "T1C1":CGSET 1,0
20 U0$="89:;":U1$="<=>?":FOR I=0 TO 3:B$=B$+
   CHR$(172+I):X$=X$+CHR$(180+I):NEXT:V=1
30 LG=16:RG=220:TG=50:BG=100:GX=0:GY=0:GV=16:BX
   =100:BY=220:BV=6:DX=0:DY=0:DV=30:R=12:X=0:Y=
   0:D=0:G=0:
```

```
40 G0$=CHR$(212):G1$=CHR$(213):D0$=CHR$(209):D1
   $=G1$
50 FOR I=1 TO 3:PALETS I,13,48,22,1-6*(I=1):NEXT:DEF
   SPRITE 6,(3,1,0,0,0)=B$:GOSUB 310:SPRITE ON
60 VX=V+RND(R):VY=V+RND(R)
70 C=RND(5):Y=Y+VY:X=X+VX*SGN(C):GOSUB 260:T=
   STRIG(0):S=STICK(0)
80 SWAP G0$,G1$:SWAP D0$,D1$:SWAP U0$,U1$:GOSUB
   310
90 IF G=-1 THEN 120
100 IF T<8 THEN 160
110 GX=BX+6:GY=BY:G=-1
120 GY=GY-GV
130 IF GY<TG THEN G=0:GX=0:GY=0:GOTO 160
140 SPRITE 7,GX,GY
150 IF ABS(GX-X-5)<6 THEN IF ABS(GY-Y+8)<8 THEN
    330
160 IF S=1 THEN BX=BX+BV:IF BX>RG THEN BX=RG
170 IF S=2 THEN BX=BX-BV:IF BX<LG THEN BX=LG
180 IF D=-1 THEN 210
190 IF ABS(BX-X)>5 THEN 240
200 DX=X+6:DY=Y:D=-1
210 DY=DY+DV:IF DY>255 THEN SPRITE 5:D=0:DX=0:DY=
    0:GOTO 240
220 SPRITE 5,DX,DY
230 IF(BY-DY)<6 THEN IF ABS(BX-DX+6)<8 THEN 360
240 IF BG<210 THEN IF RND(4)=0 THEN TG=TG+V:BG=BG
    +V
250 GOTO 70
260 IF X<LG THEN X=LG:VX=-VX
270 IF X>RG THEN X=RG:VX=-VX
```

```
280 IF Y<TG THEN Y=TG:VY=-VY
290 IF Y>BG THEN Y=BG:VY=-VY:IF BG>210 THEN 370
300 RETURN
310 DEF SPRITE 0,(1,1,0,0,0)=U0$:DEF SPRITE 5,(0,0,0,0,
    0)=D0$:DEF SPRITE 7,(3,0,0,0,0)=G0$
320 SPRITE 0,X,Y:SPRITE 7,GX,GY:SPRITE 5,DX,DY:SPRITE
    6,BX,BY:RETURN
330 FOR I=0 TO 9:SWAP X$,U0$:DEF SPRITE 0,(3,1,0,0,0)
    =U0$:SPRITE 0,X,Y:PAUSE 10:PLAY "O5F1":NEXT
340 PR=PR+10:LOCATE 15,0:PRINT "SCORE:"PR:GX=0:GY
    =0:V=1+PR/20:IF PR=100 THEN 380
350 GOTO 30
360 FOR I=0 TO 5:SPRITE 6:PLAY "O1B1":PAUSE 8:SPRITE 6,
    BX,BY:PAUSE 10:NEXT:SPRITE 5:DX=0:DY=0:D=0:B=B
    +1:IF B<4 THEN 30
370 PLAY "O1C4EC":INPUT "PRESS RETURN TO CONT!";I:
    RUN
380 PLAY "O3G3E2GO4C5A4G3ED2EC5"
390 LOCATE 0,1:FOR I=1 TO 3:FOR J=1 TO 8:PFINT " ";:
    PLAY "O1G1":NEXT:PRINT:NEXT:SPRITE OFF
400 LOCATE 7,7:PRINT "CONGRATULATIONS!!!":PALETB 0,
    21,48,48,48:END
```

键入程序后，进入 BG GRAPHIC，按图 7—31 规则绘图。绘图完毕，运行程序将显示图 7—32 的画面。

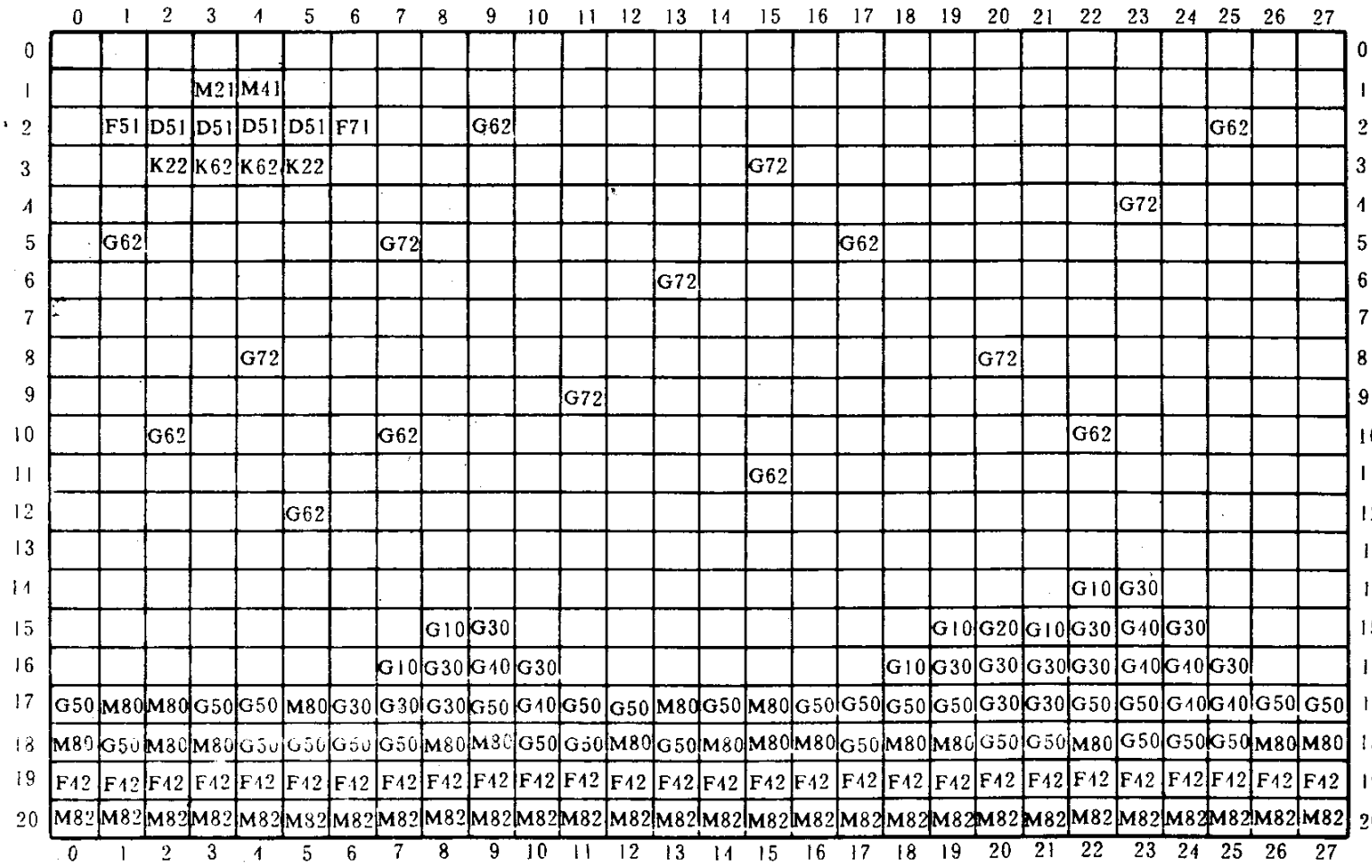


图 7—31 《UFO 驱逐战》绘图规则

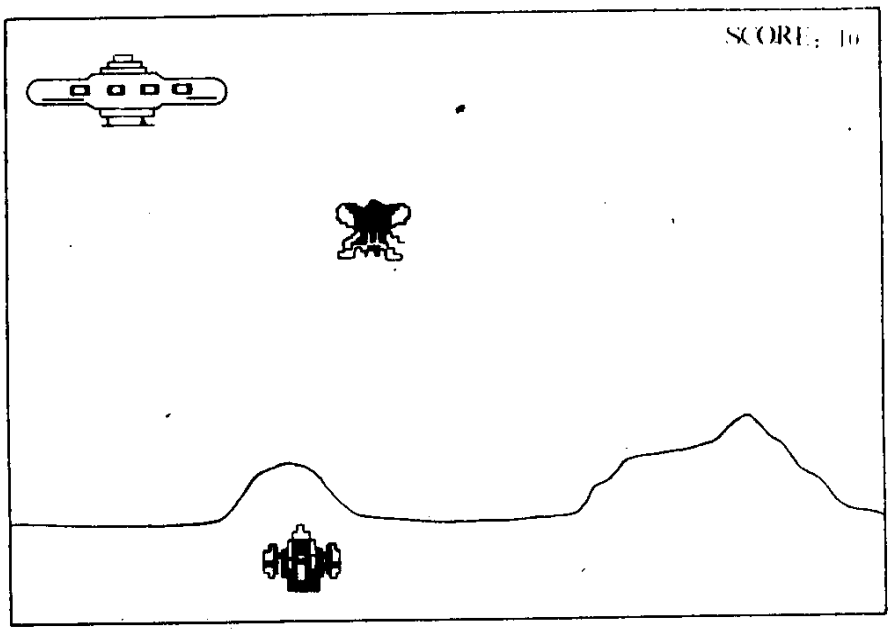


图 7—32 《UFO 驱逐战》游戏画面

现在根据《星际大战》游戏的分析方法步骤进行分析。

1. 熟悉游戏过程

该游戏为 1 人玩的游戏。由游戏者使用 I# 操纵器控制飞船左、右移动躲避苍蝇的攻击，按 A 键发射“激光刀”攻击苍蝇。苍蝇不停地变换着飞行的路线和规律，其飞行的高度逐渐降低，靠近飞船时迅速发射激光弹，命中率较高。游戏者必须敏捷地变换方位，抓住战机击毙一只只苍蝇。每击毙一只苍蝇，游戏者得 10 分。当长时间没有击毙苍蝇，使苍蝇降落到同飞船相同高度时，游戏结束，表示游戏者负，按“RETURN”键，游戏重新开始。游戏者共有四次游戏机会，若连被苍蝇击中四次则游戏结束，游戏者负，按“RETURN”键重新游戏。若游戏者连续击毙 10 只苍蝇，则使 UFO 丧失了战斗能力，只好灰溜溜地离开地球上空（画面上表现这逐渐隐去），“闪电号”飞船完成了使命，屏幕上显示出祝贺成功的字幕“CONGRATULATIONS!!!”，游戏结束。

2. 程序分段

程序 No. 7—35 是一个较典型的顺序结构程序，与 No. 7—34

相比要难读一些，很难象 No. 7—34 那样明确地划分出几个程序段。一般来说，顺序结构的程序分段方法是：首先根据 GOSUB 语句和 RETURN 语句划分出子程序；剔除子程序后，再根据 GOTO 或 THEN 语句行号对剩余程序大体分段；最后在跟踪程序运行时再具体确定各程序段。

程序 No. 7—35 中共有两个 GOSUB 语句，分别为 GOSUB 260、GOSUB 310。据此可划分出两段子程序，分别为：260～300 和 310～320。程序中 GOTO 或 THEN 的行号有 10 个，分别为：30、70、120、160、210、240、330、360、370、380。据此可暂把程序分为 10～60、70～110、120～150、160～200、210～230、240～250、300～400 共七段。

3. 跟踪程序运行，大体确定各段程序功能。

同分析模块结构程序一样，首先列出变量表和变量赋值表，见表 7—4、7—5。（可由程序的 20～40 行列出。）另外，由于该程序使用显示卡通作为动作卡通，故定义卡通的较多，可根据程序中的 50 行、310 行列出显示卡通的定义表（见表 7—6），并根据显示卡通的定位显示语句 140 行、220 行、320 行确定各显示 SP 的座标变量，一并填入表 7—6。

做完上述工作后，我们跟踪程序运行：

A. 第一周期

(1)10～40 行为初始化赋值程序，不多讨论。

(2)50 行语句设定了 SP 调色板的 1、2、3 配色号码，定义 6# 显示 SP 的卡通是太空船，调用 310 行子程序。

(3)310 行子程序。定义苍蝇为 0# 显示卡通，激光弹(2)为 5# 显示 SP、激光刀为 7# 显示 SP。并令定义的 4 个显示 SP 在屏幕上显示。各显示 SP 的显示位置为：苍蝇在(0,0)、激光弹在(0,0)、太空船在(100,200)、激光刀在(0,0)。由于在 SP 面(0,0)点在屏幕之外，所以只显太空船，其它三个卡通看不见。返回 50。

表 7—4 《UFO 驱逐战》变量表

单字符变量	双字符变量	字符串变量
X	LG	U0$
Y	RG	U1$
G	TG	B$
R	BG	X$
V	BX	G0$
C	BY	G1$
T	BV	D0$
S	GX	D1$
	GY	
	GV	
	VX	
	VY	
	DX	
	DY	
	DV	
	PR	

(4)60 行随机产生 VX、VY 的数值。70 行随机产生 C 的值,C 取值为 0、1、2、3、4,然后对 X、Y 赋值,调用以 260 行开头的子程序。

(5)260～300 行。控制 X、Y 的取值范围,由初值控制 16≤X≤200、50≤Y≤100。并使 X、Y 在取上、下限值时,令其各自的增量反向,从而使苍蝇向上飞到上限时转而向下飞,向右飞到极限时转而向左飞,反之亦然。返回 70 行。

(6)读操纵器的键值后执行 80 行。令 0#SP 苍蝇变换显示姿式,5#SP 激光弹变换形状、7#SP 激光刀变换形状,(调用 310 行后实现变换)。

通过以上讨论,基本明确了两个子程序的作用:

表 7—5 《UFO 驱逐战》变量赋值表

简单变量	字符串变量
LG＝16 RG＝220 TG＝50 BG＝100 BX＝100 BY＝220 BV＝6 GX＝0 GY＝0 GV＝16 G＝0 X＝0 Y＝0 DX＝0 DY＝0 DV＝30 D＝0 R＝12 V＝1 PR＝0	U0＄＝苍蝇(1) U1＄＝苍蝇(2) B0＝太空船(上) X＄＝爆炸(散发) G0＄＝激光刀(1) G1＄＝激光刀(2) DO＄＝激光弹(2) D1＄＝激光刀(2)

表 7—6 显示卡通定义表

SP 编号	SP 名称	所用变量
0	苍蝇	X、Y、U0＄、U1＄、
5	激光弹(2)	DX、DY、D0＄、D1＄、
6	太空船	BX、BY、B＄
7	激光刀(2)	GX、GY、G0＄、G1＄、

• 260～300 行。判断苍蝇是否越出了上、下、左、右所限定的运动范围，如果超出了运动范围则令其向相反的方向运动。同时在 290 行中还对苍蝇在 Y 方向的最大座标进行了限制，若 BG＞210，即苍蝇飞得足够低时(与太空船 Y 座标接近)则判定游戏者负，重新开始游戏。

• 310～320 行。定义 0#、5#、7#SP，并令四个卡通在各自的座

标上显示。

(7)90 行不太好理解,暂时跳过。100 行是检查游戏者是否按"A"键:若未按 A 键(未发射)则跳转 160 行;若按 A 键则执行后续程序。可见 110～150 行的程序是游戏者发射后的处理。我们先假定游戏者未按 A 键,则转 160 行。

(8)160～170 行是根据操纵器的方向键值调整飞船的 X 方向座标,以使其左、右运动,其运动速度由 BV 控制。

(9)180 行中 D＝－1 暂时也不好理解,跳过。190 行求太空船、苍蝇在 X 方向的距离,若两者距离大于 5 点则转 240 行。假定它们之间的距离大于 5 点,转 240 行。

(10)240 行。如果苍蝇在 Y 方向的最大限制范围小于 210,同时 RND(4)＝0 则调整苍蝇在 Y 方向的运动范围,使其在较低的范围内运动。然后通过 250 行返回 70 行。

由此可见,70～250 行的语句是该程序的循环主体。通过第一周期的循环,我们大体确定了如下变量的作用:

· X、Y:确定苍蝇的显示位置。

· VX、VY:调整苍蝇的运行速度。

· V:苍蝇的基本速度。

· R:控制调整苍蝇速度的最大范围。

· C:调整苍蝇在 X 方向位移的周期间隔。C 值由 RND(5)决定,当 C 值等于 0 时不调整,C＞0 则调整。

· T:操纵器的功能键值。

· S:操纵器的方向键值。

· U0$、U1$:变换苍蝇的显示姿式,以产生飞动。

· BX、BY:确定太空船的显示位置。

· BV:太空船的运动速度。

· LG、RG、TG、BG:苍蝇的运动范围。

· DX、DY:激光弹的显示座标。

• D0$、D1$：变化激光弹的形状。

• GX、GY：激光刀的显示座标。

• G0$、G1$：变化激光刀的形状。

B. 第二周期

(1)90 行。假定 G≠－1，T＝8，则执行 110 行。

(2)110 行。在 110 行对激光刀的显示座标变量 GX、GY 进行赋值，其座标等于太空船的座标，并令 G＝－1。然后在 110 行使 GY 减少 16 点。这相当于按 A 键后，太空船发射子弹，并使子弹向上运动。

(3)130 行。GY＜TG 对应于子弹在 Y 方向的座标小于苍蝇的上限，相当于未击中苍蝇，则令 G＝0、GX＝0、GY＝0 跳转 160 行。130 行的作用就是未击中苍蝇时，令子弹消失，G＝0。可见，变量 G 是太空船的发射标志，若发射则令 G＝－1，当程序执行到 90 行即转 120 行，只令子弹上行，不能发射。只有当子弹运行完毕才能进行下一次发射。如果 GY＞TG 则执行 140 行程序令 7#SP 激光刀在新的座标上显示。

(4)150 行。150 行的作用很明显，即只要激光刀和苍蝇在 X 方向的距离小于 6，同时在 Y 方向的距离小于 8 则表示苍蝇被击中，转 330 行。

(5)330 行。330 行把爆炸的数据 X$ 和苍蝇的数据 U0$ 交替赋给 0# 显示卡通，从而产生苍蝇发生爆炸的画面，同时发声。爆炸画面交替闪烁 10 次后执行 340 行，给游戏者加分，显示得分，改变速度，由 350 行返回 30 行继续游戏。

(6)若未击中苍蝇则执行 160、170 行。

(7)假定 D≠－1、|BX－X|＜5 则执行 200 行。

(8)200 行。该行对 5#SP 激光弹的显示座标变量 DX、DY 赋值，其值为苍蝇的当前座标值，并令 D＝－1，这相当于苍蝇向飞船发射。D 为苍蝇发射的标志变量，发射 D＝－1。

(9)210 行令苍蝇发射的子弹向下运行。若子弹在 Y 方向的座标大于 255 表示没有击中飞船，则令 5#SP 消失，令 D＝0，转 240 行。若子弹在 Y 方向的座标小于 255 则令 5#SP 显示，由 230 行判断是否击中飞船，若击中则转 360 行。

(10)360 行。令飞船闪 6 次并发声，记录飞船被击中的次数，若小于 4 次则转 30 行继续游戏，若等于四次则由 370 行通过按回车键重新游戏。

(11)若太空船未被击中则通过 240 行调整苍蝇的飞行高度后，跳转 70 行进行下一循环。

通过第二周期的运行我们确定了剩余变量的作用：

·GV：太空船发射的激光刀的运行速度。

·G：太空船发射标志。发射后 G＝－1，子弹运行完毕 G＝0。G＝0 可进行第二次发射。

·DV：苍蝇发射的激光弹的运行速度。

·D：苍蝇发射标志。发射后 D＝－1，子弹运行完毕 D＝0。D＝0 可进行第二次发射。

·PR：游戏者的得分，每击中一个苍蝇得 10 分。

另外在 340 行中有当 PR＝100 时转 380 行，意味着若游戏者击中 10 个苍蝇则胜，转 380 行进行胜利后的处理。380～400 行为胜利后的处理程序。首先奏一段音乐；然后令 UFO 逐渐消失；最后打印祝词和显示红色背景，结束。

4. 综合分析

通过以上的分析讨论，该程序各语句的作用已大体明确，现综述如下：

(1)程序中第 20～40 语句用来定义字符串和给一些变量赋值。

·20 行定义了字符串 U0$、U1$、B$、X$。在后面的 DEF SPRITE 指令中分别用于定义显示卡通苍蝇、太空船和爆炸。

·40 行定义字符串 G0$、G1$、D0$、D1$。在后面的 DEF SPRITE 指令中分别用于定义激光刀和激光弹。

·30 行给一些变量赋值。LG、TG 分别定义苍蝇和太空船左、右运动范围；TG、BG 定义苍蝇的上、下运动范围。GX、GY 是激光刀的座标；GV 是激光刀运动的速度。DX、DY 是激光弹的座标，DV 是激光弹运动的速度。BX、BY 是太空船的座标，BV 是太空船运动的速度。X、Y 是苍蝇的座标，它的运动速度由第 60 行的随机函数控制。

(2)50 行调用 310 行的子程序，定义 0#SP 为苍蝇、5#SP 为激光弹、6#SP 为太空船、7#SP 为激光刀。

(3)60 行随机产生苍蝇在 X、Y 方向的运动速度 VX、VY。

(4)程序的主体是 70～250 行的循环体。

·70 行首先调整苍蝇的位置座标 X、Y，然后调用 260 行的子程序，判断苍蝇是否越出了上、下、左、右所限定的运动范围，若超出运动范围，则令其向相反方向运动。再从操纵器取得方向键值 S 和功能键值 T。

·80 行交换字符变量 G0$ 和 G1$、D0$ 和 D1$、U0$ 和 U1$，再次调用 310 行子程序。交换字符串变量的目的是使苍蝇、激光刀、激光弹的两种姿式交替出现，以产生逼真的动感。

·90～140 行控制激光刀的运动。当太空船发射激光刀后，在 110 行令发射标志变量 G＝－1，使以后程序再次循环执行到 90 行时转到 120 行，令激光刀的 Y 方向座标递减，这样就使激光刀连续向上运动，直到其 Y 座标等于 TY，由第 130 行令 G＝0，允许再次发射。

·150 行判断激光刀是否击中苍蝇，若击中则跳转 330 行。

·160～170 行根据操纵器的方向键值调整太空船的 X 座标 BX，控制太空船左、右运动。

·180～220 行控制激光弹的运动，其原理同激光刀。

• 230 行判断激光弹是否命中太空船，若命中则转 350 行令太空船发生闪烁，且令太空船被击中次数的记忆变量 B 加 1，如 B＝4 则游戏结束，屏幕上出现“PRESS RETURN TO CONT!”提示(按回车键继续)，待游戏者按“RETURN”键后，游戏重新开始。

• 250 行控制程序跳转到 70 行，使程序循环执行。

(5)330～350 行完成当太空船发射的激光刀击中苍蝇后，令苍蝇爆炸；显示得分和累计得分；令苍蝇的运动速度增大；同时检测是否击中 10 个苍蝇，若击中 10 个则转驱逐 UFO 成功处理。

(6)380～400 行完成胜利后的奏乐，令 UFO 消失、令 SP 消失，打印祝贺词和改变底背景色调等多项操作。

5. 技巧小结

(1)程序 No. 7—35 使用显示 SP 代替了动作 SP，从而拓宽了动作 SP 的应用领域。我们知道，若使用 DEF MOVE 语句定义 SP 只能使用系统中固有的 16 种卡通，而使用 DEF SPRITE 语句可定义我们喜爱的任意卡通，从而使自己设计的卡通应用于游戏之中，如定义带翅膀的玛丽飞行作战等等。因此，我们可从该程序中获得显示卡通运动控制、两卡通相撞的判断及卡通的显现与隐藏等方面的处理技巧。

(2)从 60、70 行中学习卡通随机运动的控制技巧及运动速度的调整技巧。

(3)学习卡通发射后的命中与否的处理技巧。

(4)学习 F BASIC 程序与 BG GRAPHIC 绘图的结合技巧等。

7.7 游戏程序介绍

除以上各章节介绍的游戏示例之外，本节再介绍几个游戏实例，以飨读者。

7.7.1 智能游戏《超级记忆》

例 7.10 《超级记忆》

《超级记忆》是一个单人玩的测验记忆力的游戏。游戏者必须记住彩色方块在屏幕上、下、左、右闪动的顺序，然后使用键盘的上、下、左、右键输入该顺序，若输入的顺序正确则闪一下你输入的彩色方块，进行第二轮测验，每猜对一次，闪动的方块数增加一个，直到方块数增加到最大值，若仍然记忆正确，则打印“VERY GOOD!”结束运行，若记忆的顺序不正确则重新显示一遍，由你继续输入顺序，直到猜对再进行下一轮测验。

程序清单如下：

```
5 REM "No. 7—36,SUPER MEMORY"
10 CLS:LOCATE 6,10:PRINT "PLEASE SELECT 5～20";:INPUT
   MAX
20 VIEW:I=Z:X=Y:C=0
30 PP$ ="CFGE"
40 FOR K=1 TO 5:Z$ =Z$ +CHR$ (254):NEXT
50 N$ ="          "
60 DIM PL(MAX),PX(3),PY(3),C(3)
70 PX(0)=16:PY(0)=8:C(0)=2
80 PX(1)=0:PY(1)=8:C(1)=4
90 PX(2)=8:PY(2)=0:C(2)=6
100 PX(3)=8:PY(3)=16:C(3)=8
110 REM "MAIN PROGRAM"
120 PL(Z)=RND(4)
130 FOR I=0 TO Z
140 X=PX(PL(I)):Y=PY(PL(I)):C=C(PL(I))
150 PALETB 0,13,13,13,13
160 GOSUB 400
170 PLAY "T2O4"+MID$ (PP$ ,PL(I)+1,1)+"3"
180 GOSUE 450
```

```
190 PALETB 0,13,22,39,2
200 PAUSE 10:NEXT
210 I=0
220 LOCATE 9,10:PRINT "YOU"
230 A$=INKEY$:IF A$=" "THEN 230
240 IF A$<CHR$(28) OR A$>CHR$(31) THEN PLAY
    "T1O1C1CC":GOTO 230
250 IF(ASC(A$)-28)<>PL(I)THEN PLAY "T2O5C2RFRE":
    LOCATE 9,10:PRINT "      ":GOTO 130
260 PALETB 0,13,13,13,13
270 X=PX(PL(I)):Y=PY(PL(I)):C=C(PL(I))
280 GOSUB 400
290 PLAY "T2O4"+MID$(PP$,PL(I)+1,1)+"3"
300 GOSUB 450
310 PALETB 0,13,22,39,2
320 I=I+1:IF I<=Z THEN 220
330 Z=Z+1:IF Z>MAX THEN 360
340 LOCATE 9,10:PRINT "      "
350 PAUSE 50:GOTO 110
360 REM "END"
370 CLS:CGSE 1,1:PRINT "VERY GOOD!"
380 END
400 FOR J=0 TO 5
410 LOCATE X,Y+J:PRINT Z$:NEXT
420 PALETB 0,13,C,C,C
430 RETURN
450 PALETB 0,13,13,13,13
460 FOR J=0 TO 5
470 LOCATE X,Y+J:PRINT N$:NEXT
480 RETURN
```

游戏背景按图 7—33 绘制。

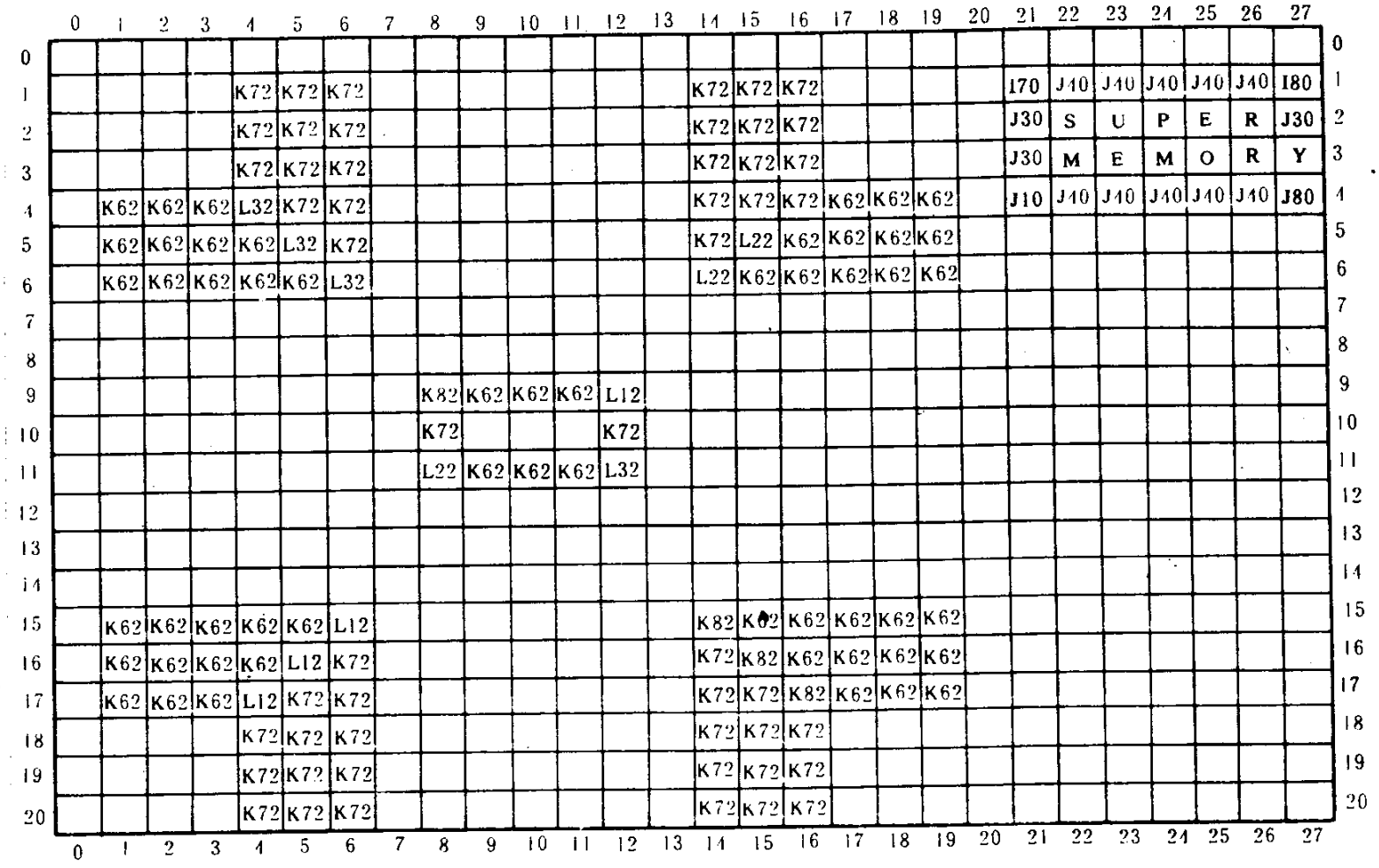


图 7—33 《超级记忆》游戏背景绘图规则

绘图完毕，运行程序。首先显示选项提示

"PLEASE SELECT 5～20"? □

要求游戏者选择每次最多闪动彩块的个数(即难度选择。数值越大，难度越大。)，我们选择最低等级，键入"5"，↙则显示如图7—34的游戏画面。

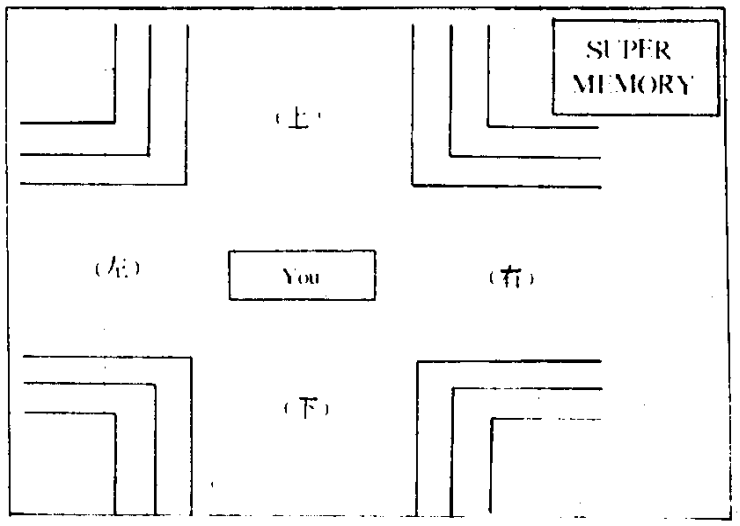


图7—34 《超级记忆》游戏画面

画面上画出了上、下、左、右四个显示区，彩色方块将在这四个区中闪烁。

第一次只闪烁一个彩块，如：左"上"区闪烁。随后中心方框中显示出"YOU"的字符，表示"你现在可以输入"。我们若按键盘上的[↑]键，则彩块又在"上"区闪烁一次，然后中心方框中的"YOU"消失，稍停 一会(第一轮测验结束)，进行第二轮测验。这一次要有两个彩块闪烁，假定为"上"、"下"，如果我们先按了[↓]键则发声"143"，又按"上、下"区顺序闪烁一次。如果我们按[↑]、[↓]则进入第三轮测验……。在第五轮测验中，若彩块的闪烁顺序是"上、下、上、左、左"，如果我们依次按[↑]、[↓]、[↑]、[←]、[←]，则清屏，显示"VERY GOOD!"，停机。

例7.11 程序分析：

该程序较简单，简要分析如下：

1. 10～100 行为游戏选项和初始化赋值程序。其中 10 行打印选项提示，由游戏者选择难度；30 行定义每个区间闪烁彩块时的伴音；40 行定义 Z$ 字符变量为由 M7 号背景图案组成的长方条（由五块组成），这个长方条在后面每 400 行开始的子程序中用于拼出闪烁彩块；50 句定义 N$ 为 5 的空格，这个字符串在后面的第 450 行开始的子程序中用于使彩块消失，以达到闪烁的目的；70～100 行确定彩块在上、下、左、右四个区中的显示座标和颜色，这四个座标分别为右(16,8)、左(0,8)、上(8,0)、下(8,16)。由 PX 数组记录 X 座标、PY 数组记录 Y 座标，C 数组记录色块的颜色。

2. 120～200 行的语句控制彩块闪烁。在这段程序中变量 Z 控制每场测验中彩块连续闪烁的次数，Z 的取值由 330 行决定。第 120 行的语句 PL(Z)＝RND(4)是使 PL 数组为不可预测数组，从 PL(0)到 PL(MAX)的值都是事先不可预知的，从而使每场彩块闪烁的位置顺序成为事先不可预测。在 160 行调用 400 行子程序是使彩块显示，同时发声；180 行调用 450 行子程序是使彩块消失，由此使屏幕上出现闪烁彩块。

3. 彩块闪烁结束后，由 220 行使画面中心的长方框中出现“YOU”，提示游戏者按键回答。

4. 230 行读取游戏者所按下的键符，由 240 行判断是不是 ↑、↓、←、→ 键，若不是则发声“111”，要求游戏者重新回答。

5. 250 行判断游戏者按下的键是否正确。若不正确则返回 130 行重新闪烁一遍；若正确则通过 260～310 行使该键所对应的彩块再闪烁一次。然后经 320 行转 250 行读取本场测验中的下一个彩块的回答键符（假定本场彩块在两个以上。若仅有一个彩块，即 Z＝1，则进行第二场测验）。直到本场测验的彩块全部正确回答完毕，则由 330 行令 Z＝Z＋1，即使下一场测验的闪烁彩块数增加一个，返回 120 行进行下一场测验。

6. 若 Z＞MAX，即每场闪烁的彩块次数达到游戏选择的难度

值时，则游戏结束，转 360 行，清屏，打印“VERY GOOD!”结束运行。

读者可把 380 行改写为

```
380 PRINT "INTO NEXT……":MAX=MAX+1:CLEAR : GOTO
    20
```

那么，计算机会自动把难度等级升一级，重新进行测验游戏。

本游戏可在所有 F BASIC 版本系统中运行。

7.7.2 智能游戏《翻纸牌》

例 7.12 《翻纸牌》

《翻纸牌》也是一个测验记忆力的游戏，玩好游戏的关键是记住已翻过的扑克牌的数码和花色，使每次翻出的两张牌数码、花色相同。

游戏玩法：

这是一个双人玩的游戏，游戏者使用操纵器的方向键移动选牌光标选定纸牌，用 A 键翻牌，若翻开的两张牌数码、花色相同则胜，加 10 分后，继续翻牌。若翻错了则翻过的牌又恢复原样，由对方翻牌。

《翻纸牌》游戏程序见 No. 7—37。

```
5 REM "No. 7—37,TURN CORDS"
10 VIEW:CGEN 3:SPRITE ON
20 I=J:A=C:S=T:X=-1:Y=-1:ED=P:CX=CY:T0=T1:TT
   =0
30 FOR I=0 TO 3:N$=N$+CHR$(254):NEXT:M0$=CHR
   $(243)+CHR$(247):M1$=CHR$(245)+CHR$(248)
40 DIM D(5,5),PT(17),PR(1)
50 FOR I=0 TO 5:FOR J=0 TO 5:LOCATE I*4+1,J*3+1:
   PRINT M0$:LOCATE I*4+1,J*3+2:PRINT M1$
60 A=RND(18):IF PT(A)=2 THEN 60
70 D(I,J)=A:PT(A)=PT(A)+1:NEXT:NEXT
```

```
80 PALETS 0,13,18,34,2:PALETS 1,13,20,36,4:PALETS 2,13,
   22,38,6:PALETS 3,13,2,25,54
90 DEF SPRITE 0,(3,1,0,0,0)=N$:DEF SPRITE 5,(3,1,0,0,0)
   =N$
100 LOCATE 25,9:PRINT "L.":LOCATE 25,12:PRINT "R."
110 SPRITE 0,CX*32+24,CY*24+31:T0=-1:T1=-1:TT=7:
    GOSUB 170:SWAP T0,T1:X=CX:Y=CY:TT=6:GOSUB 170
120 IF T0<>T1 THEN 150
130 PLAY "T2O4C1EGO5C6":ED=ED+1:IF ED=18 THEN 300
140 PR(P)=PR(P)+10:LOCATE 24,10+P*3:PRINT PR(P):
    SPRITE 6:SPRITE 7:GOTO 110
150 PLAY "O1T2D5EC"
160 D(X,Y)=T1:D(CX,CY)=T0:LOCATE CX*4+1,CY*3+1:
    PRINT M0$:LOCATE CX*4+1,CY*3+2:PRINT M1$:LO-
    CATE X*4+1,Y*3+1:PRINT M0$:LOCATE X*4+1,Y*3
    +2:PRINT M1$:P=1-P:GOTO 110
170 SPRITE 5,216,95+24*P:PAUSE 5:SPRITE 5:PAUSE 5:S=
    STICK(P):T=STRIG(P)
180 IF S=1 THEN CX=CX+1:IF CX>5 THEN CX=0
190 IF S=2 THEN CX=CX-1:IF CX<0 THEN CX=5
200 IF S=4 THEN CY=CY+1:IF CY>5 THEN CY=0
210 IF S=8 THEN CY=CY-1:IF CY<0 THEN CY=5
220 IF S=0 THEN 240
230 SPRITE 0,CX*32+24,CY*24+31
240 IF T<8 THEN 170
250 SWAP D(CX,CY),T0
260 IF T0=-1 THEN SWAP D(CX,CY),T0:GOTO 170
270 SPRITE 0:PLAY "O5T1C1C":PAUSE 10:LOCATE CX*4+1,
    CY*3+1:PRINT "      ":LOCATE CX*4+1,CY*3+2:
    PRINT "      "
280 DEF SPRITE TT,(T0/6,1,1,0,0)=CHR$(48+T0-T0/6*6):
```

```
    SPRITE TT,CX*32+24,CY*24+31
290 RETURN
300 PRINT "TRY AGAIN!!"
310 T=STRIG(0):IF T=0 THEN 310
320 IF T=1 THEN RUN
330 IF T=2 THEN END
340 GOTO 300
```

键入程序后，按图 7—35 规则键控绘图。

运行程序将显示图 7—36 的游戏画面。在左上角的第一张牌上有一绿色光标，最右边“L.”上有一闪烁的绿色光标，闪烁光标指示哪一个游戏者游戏。现在闪烁光标在“L.”上，表示由左边的游戏者，即 I# 操纵器游戏。按动 I# 操纵器的方向键可控制绿色光标上、下、左、右移动以选择纸牌，选定纸牌后，按 A 键即可把选定的牌翻过来。然后再按方向键选择另一张牌翻过来。若翻开的两张纸牌花色、数码相同，则在“L.”下显示 10 分，且所翻开的两张纸牌在屏幕上消失，可继续翻牌。若花色、数码不同，则恢复纸牌的背面，此时，游戏者应记住此位置下纸牌的花色、数码，以便下次轮到游戏时，可以准确地翻出两张相同的纸牌。现在，闪烁绿色光标移到“R.”上，表示由右边的游戏者即 Ⅱ# 操纵器游戏。如此，轮流翻牌，直到 36 张纸牌全部配对消失，游戏结束。打印“TRY AGAIN!!”，询问“再尝试一次”。按“START”键重新游戏；按“SELECT”键则结束游戏。

程序说明：

1. 变量和数组的作用

20 行定义的变量 X、Y 和 CX、CY 分别用于表示翻开的两张纸牌的座标；ED 用于记录已经翻对过的纸牌的对数，如果 ED＝18 则游戏结束；变量 P 是操纵器选择变量，P＝0，I# 操纵器有效，P＝1，Ⅱ# 操纵器有效；TT 是翻开的两张牌的编号，TT＝6、7，程序

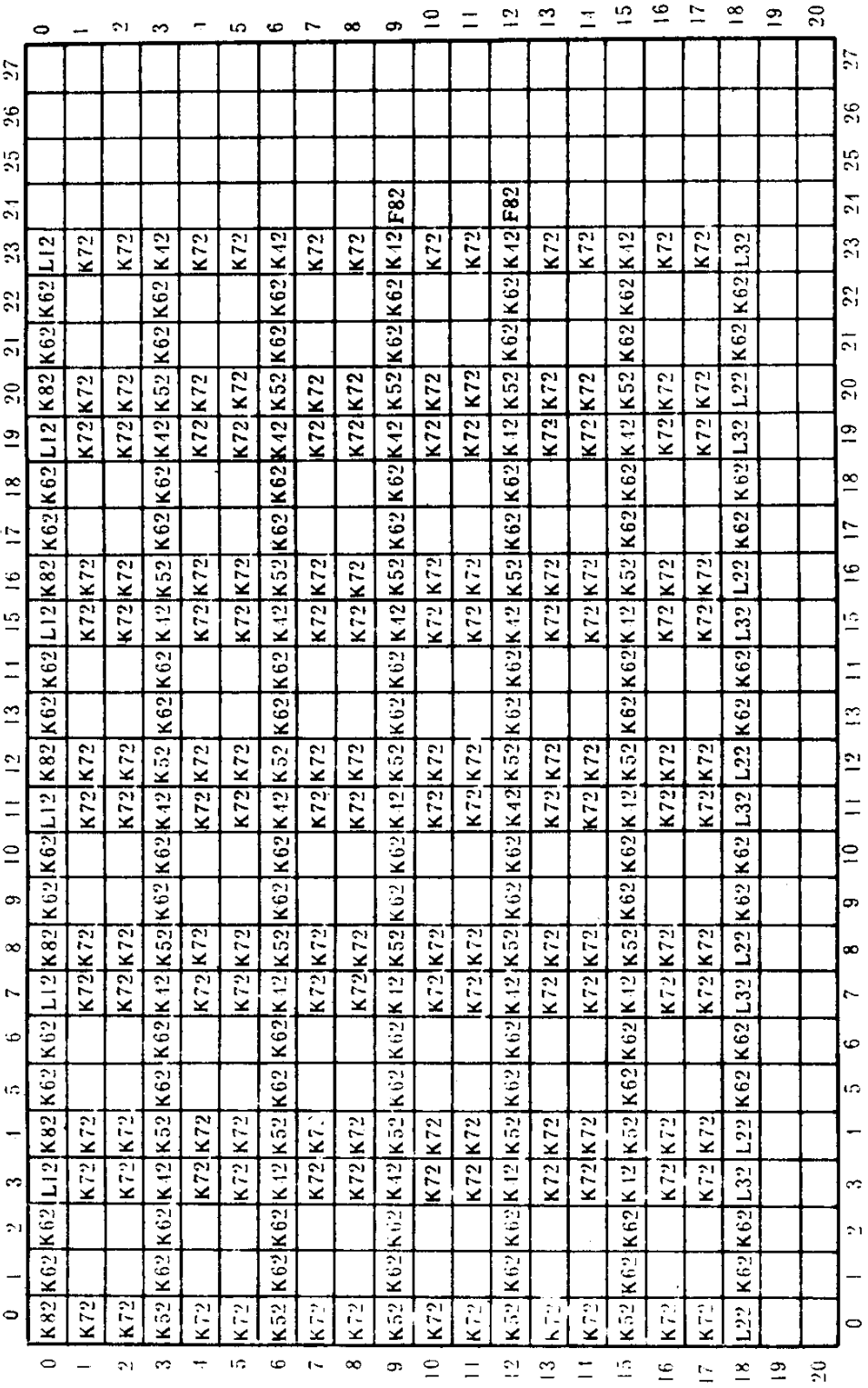


图 7—35 《翻纸牌》游戏画面绘图规则

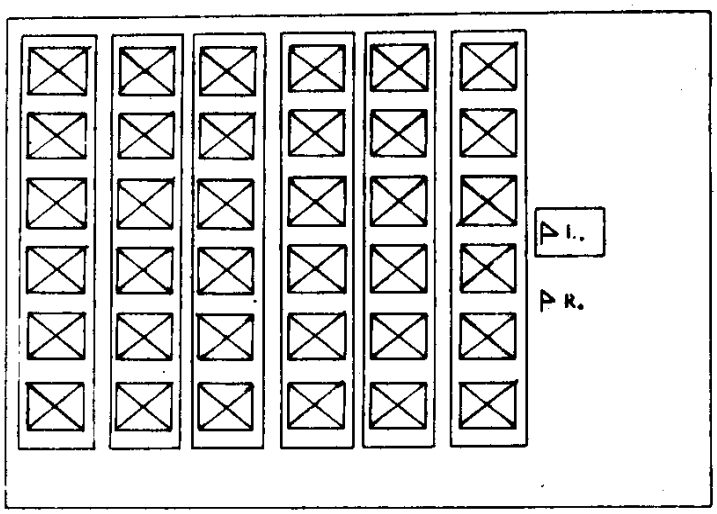


图 7—36 《翻纸牌》游戏画面

中把被翻开牌的花色、数码定义为显示卡通，第一张为 7# 显示 SP，第二张为 6#。

30 行定义的 N $ 用来定义显示 SP0# 和 5#，0# SP 用于选择纸牌的光标，5# 用于指示哪一个游戏者翻牌的闪烁光标。M0 $、M1 $ 用于拼出纸牌背面的图案。

40 行定义的数组 D(5,5)是一个 6×6 的数组，用它来记忆 36 张纸牌的花色号码。PR(1)数组用于记录得分：PR(0)用来记录 I# 操纵器的得分，PR(1)记录 Ⅱ# 操纵器的得分。PT(17)有 18 个变量，用于控制相同的纸牌只能有两张。

2. 50～70 行打印 36 张牌的背面，并随机产生 36 张牌的点数，使用 PT 数组控制相同号码的纸牌只能有两张。

3. 90 行定义 0#、5# 显示 SP 为两个光标。100 行在屏幕右边打印“L.”和“R.”字样(左、右)，当光标在“L.”上闪烁时，I# 操纵器有效；光标在“R.”上闪烁时，Ⅱ# 操纵器有效。

4. 110～160 行语句是程序的主体。它控制着 I#、Ⅱ# 操纵器的选牌、翻牌、判断所翻开的两张牌是否相同、计算得分和判断游戏是否结束。

· 110 行的作用是控制两次翻纸牌。开始时使光标出现在

CX,CY 所指示的纸牌的位置，令 T0＝T1＝－1，TT＝7，调用 170 行开头的子程序。

• 170 行～290 行子程序的作用：显示闪烁光标；(170 行)读操纵器键值；根据键值改变指示光标的位置(180～230 行)；翻开纸牌(240～280 行)。

• 第一次调用 170 行子程序时，将所选择的纸牌的号码在第 250 行交换给 T0，并于第 270、280 行在(CX,CY)的位置将纸牌翻开。

• 从子程序返回后，交换 T0 和 T1，这样所翻开的第一张纸牌的号码就交换给 T1，而 T0＝－1，然后再用 X,Y 记下第一张纸牌的位置座标 CX、CY 记录下来。

• 第二次调用 170 行子程序时，第 250 行又将所翻开的第二张纸牌的号码交换给 T0，并于第 270、280 行翻开第 2 张纸牌。

此时，经过两次调用 170 行子程序，使已翻开的两张纸牌的号码分别记于 T1 和 T0，两张纸牌的位置座标分别记于(X,Y)和(CX,CY)。

• 第 260 行语句的作用是对两次翻同一张牌或在纸牌已消失的地方禁止翻牌。

5. 在两张纸牌都翻开后，由 120 行语句判断两张牌的号码是否相同。若相同，首先判断所有纸牌是否都已翻开(即 ED 是否等于 18)，若都已翻开则游戏结束；若 ED≠18 则给游戏者加分，并显示得分，最后使翻开的两张纸牌消失(140 行)。

若两张纸牌不同时则恢复两张纸牌的号码，恢复纸牌的背面，换另一个操纵器来选牌。这些处理是由 160 行语句完成的。

6. 300～340 行是结束处理程序。

程序 No. 7—37 可大体分为四段：10～100 行为初始赋值、预处理程序；110～160 行为主控程序(其中 130、140 行为两张纸牌相同的处理；150、160 行为两张纸牌不同的处理。)；170～290 行为

翻牌处理子程序;300～340 行为结束处理程序。

本游戏可在所有 F BASIC 版本系统中运行。

7.7.3 赌博游戏《乌龟赛跑》

例 7.13 乌龟赛跑

乌龟赛跑是一个由计算机自动进行的赌博类型的游戏。

游戏方法:

本游戏最多可以五个人一起玩。游戏中有五个乌龟进行赛跑,比赛前,由游戏者进行预测几号乌龟可能获胜则选几号乌龟,如预测正确,则根据该号乌龟的倍率计算出成绩,打印在屏幕上。源程序见 No. 7—38

```
10 REM "No. 7—38,TORTOISE RACE"
20 CLS:CGSET 1,2
30 I=K:A=J:BT=5:MAX=3:PP=PR:W=-1
40 PLAY "O4T1":P$="C2D2E2F2G2A2B2C2"
50 DIM AX(4),AY(4),Q(4),C(5),P(5,1)
60 INPUT "PLAYERS";PL:IF PL=0 OR PL>5 THEN PLAY
   "C1CC":CLS:GOTO 60
70 VIEW:LOCATE 13,0:PRINT "TOR.":LOCATE 13,2:PRINT
   "MUL."
80 FOR I=0 TO 4:Q(I)=RND(MAX)+BT:C(I)=MAX+BT-Q
   (I)
90 DEF SPRITE 1,(0,1,0,0,0)=CHR$(184)+CHR$(185)+CHR
   $(186)+CHR$(187)
100 LOCATE 17+2*I,0:PRINT I+1
110 LOCATE 17+2*I,2:PRINT C(I)
120 AX(I)=220:P(I,0)=-1:NEXT
130 FOR I=0 TO PL-1
140 LOCATE 0,21:PRINT "PLAYER"I+1
150 INPUT "TOR. NO.?",M:IF M=0 OR M>5 THEN PLAY
    "C1CC":GOTO 140
```

```
160 P(I,0)=M-1:NEXT
170 LOCATE 0,21:PRINT "                              ":PRINT "
                    "
180 SPRITE ON
190 FOR I=0 TO 4:A=RND(RND(Q(I)))
200 AY(I)=71+24*I:AX(I)=AX(I)-A*2
210 IF AX(I)<0 THEN AX(I)=0
220 PLAY MID$(P$,A*2+1,2):SPRITE I,AX(I),AY(I)
230 IF AX(I)<50 THEN GOSUB 340
240 NEXT
250 K=K+1:IF K=20 OR K=50 THEN GOSUB 270
260 GOTO 180
270 FOR I=0 TO 4:LOCATE 10,10:PRINT "CHANGE!"
280 A=RND(8):IF A=7 THEN Q(I)=9:GOTO 320
290 IF A=6 THEN IF RND(5)=4 THEN Q(I)=0:GOTO 320
300 IF A<5 THEN 320
310 Q(I)=11-Q(I)
320 NEXT
330 LOCATE 10,10:PRINT "                    ":RETURN
340 IF I=W THEN RETURN
350 IF W<>-1 THEN 370
360 W=I:RETURN
370 PLAY "T6CEDFEGC2T1":LOCATE 10,6:PRINT "First"W+1
    "Second"I+1
380 SPRITE OFF
390 FOR J=0 TO PL-1:PR=0
400 IF P(J,0)=W THEN PR=2
410 IF P(J,0)=I THEN PR=1
420 PP=PR*C(P(J,0)):LOCATE 5,9+J*2:PRINT "PLAYER"J
    +1" SCORE"PP
430 P(J,1)=P(J,1)+PP:NEXT
```

```
440 LOCATE 16,22:PRINT "TRY AGAIN?";:A$=INKEY$(0):
    IF A$="N" THEN 460
450 K=0:W=-1:GOTO 70
460 CLS:FOR J=0 TO PL-1:LOCATE 4,9+J*2:PRINT "PLAY-
    ER"J+1" TOTAL SCORE";P(J,1):NEXT
470 LOCATE 5,22:PRITN "------END------":PLAY
    "CDGAC":END
```

输入程序后，按图 7—37 规则键控绘图。运行程序。在屏幕上方显示几人游戏的提问：

"PLAYERS"? □

若 4 人游戏则键入"4"，回车后，即显示图 7—38 的游戏画面。上面两行提示的意思是：第一行是乌龟编号，第二行是该编号下对应的倍率。一般情况下倍率低的乌龟取胜的可能性较大，但也有特殊情况。屏幕的左下角也有两行提示：

PLAYER 1

TOR. No. ? □

意思是："第 1 号游戏者，请输入押注的乌龟号码"。1 号游戏者输入选定的乌龟号码后，回车，则显示

PLAYER 2

TOR. NO? □

要求第 2 号游戏者输入选定的乌龟号码……。当 4 名游戏者的选项输入完毕后，比赛开始。五个乌龟各自以不同的速度向终点奔跑。比赛中间，屏幕上将闪烁"CHANGE"字样(中文意思是变化)，这时各乌龟的速度又发生了变化，原来有希望获胜的乌龟，可能落在最后；而冲线无望的乌龟可能跑在最前面；也可能跑得快的更快，跑得慢的更慢。它们的速度随机产生，随机改变，没有规律。好！第一名冲线了，是 1 号乌龟。冲线后，它继续向前跑，直到第二名又冲线(假定 4 号乌龟)则比赛结束。在屏幕中间显示

First 1 second 4

“第一名是 1 号，第二名是 4 号”。随后显示各游戏者的得分，假定 1 号游戏者选 2、2 号游戏者选 4、3 号游戏者选 1、4 号游戏者选 5，则打印如下结果：

```
PLAYER  1  SCORE  0
PLAYER  2  SCORE  3
PLAYER  3  SCORE  2
PLAYER  4  SCORE  0
```

由于 1 号乌龟的倍率是 1，第一名得 2 分，所以 3 号游戏者的得分为 2×1=2 分；而 4 号乌龟的倍率是 3，第二名得 1 分，所以 2 号游戏者的得分是 1×3=3 分。

在屏幕的下部还显示一行

```
TRY AGAIN? □
```

意思是“再尝试一次吗?”，按除“N”键外的任一键则继续游戏；若结束游戏则按“N”键，这时将清屏，打印各游戏者的累计得分。

本程序顺序编写，简单易懂，仅简要说明如下，读者自行详细分析。

1. 10～60 行完成初始化和游戏人数选项。

2. 70～120 行完成随机产生各乌龟的倍率，并把乌龟号码和对应的倍率打印在屏幕上。

3. 130～170 行完成游戏者对乌龟号码的选项。

4. 180～260 为主体程序。控制 5 个乌龟运动。第一名冲线的乌龟由 230 行调用 340 行开始的子程序记录第一名。引入变量 K 控制变速的时间间隔。

5. 270～330 行为乌龟变速子程序。

6. 340～360 行为记录第一名子程序。

7. 370～430 行打印名次、游戏者得分和累计各游戏者得分。

8. 440～470 行为结束处理程序。其中 460 行打印各游戏者的

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60	D60
1	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62	D62
2	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52	G52
3																												
4																												
5	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K62	K62	K62	K62
6				J30																						J32	1	
7	1			J30																						J32		
8	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K62	K62	K62	K62
9				J30																						J32	2	
10	2			J30																						J32		
11	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K62	K62	K62	K62
12				J30																						J32	3	
13	3			J30																						J32		
14	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K62	K62	K62	K62
15				J30																						J32	4	
16	4			J30																						J32		
17	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K62	K62	K62	K62
18				J30																						J32	5	
19	5			J30																						J32		
20	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K60	K62	K62	K62	K62

图 7—37 《乌龟赛跑》游戏绘图规则

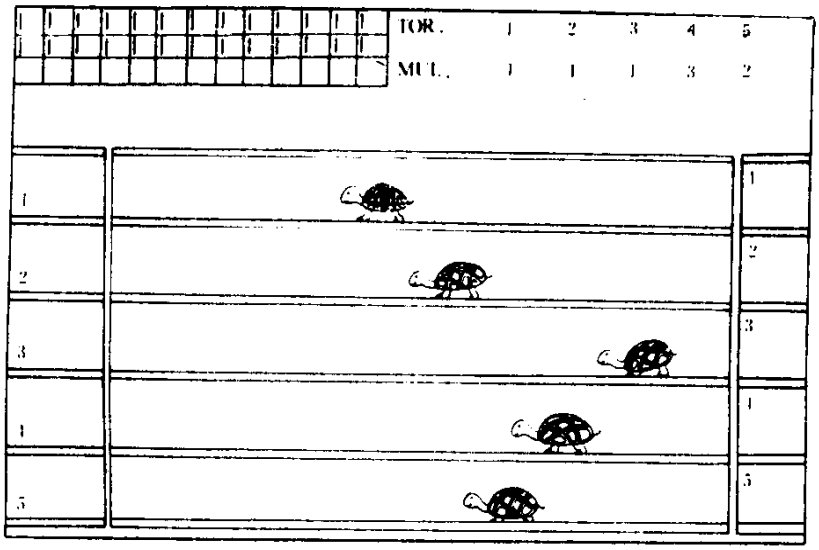


图 7—38 《乌龟赛跑》游戏画面

积分。

本程序适用所有 F BASIC 版本。

7.7.4 多版游戏《玛丽采宝》

例 7.14 《玛丽采宝》

《玛丽采宝》是一个多场游戏。只要你玩得好，可以连续打 200 关，每关的画面随机产生，各不相同，永不重复，能给你带来无比的新鲜感和无穷的乐趣。

源程序见 No. 7—39。本例是程序 No. 7—31 的具体应用。

```
5 REM "No. 7—39"
10 REM "MARY PICKS THE TREASURES"
15 CLEAR:CGSET 1,0:SPRITE ON
20 ON ERROR GOTO 445
25 K$=CHR$(24):R$=CHR$(25):H$=CHR$(26):U$=
   CHR$(27)
30 DEF SPRITE 0,(0,1,0,1,0)=R$+K$+U$+H$
35 K$=CHR$(12):R$=CHR$(13):H$=CHR$(14):U$=
   CHR$(15)
40 DEF SPRITE 1,(0,1,0,1,0)=R$+K$+U$+H$
```

```
45 DEF SPRITE 2,(0,1,0,0,0)=K$+R$+H$+U$
50 K$=CHR$(20):R$=CHR$(21):H$=CHR$(22):U$=
   CHR$(23)
55 DEF SPRITE 3,(0,1,0,0,0)=K$+R$+H$+U$
60 R$=CHR$(215):H$=CHR$(211):U$=CHR$(192):DIM
   Q(3),D(3)
65 REM "BEGIN"(开始)
70 T=300:W=0:NN=1:X=116:Y=127:H=0:POSITION 7,X,Y:
   GOSUB 435:V=0:GOTO 260
75 REM "THE END CONTROL"(结束控制)
80 T=T-1:IF TMOD 10=0 THEN LOCATE 14,21:PRINT "    ":
   IF T=0 THEN 320
85 IF R=10 THEN CUT 0,1,2,3,4,5,6,7:PLAY "T2:T2:T2":GO-
   SUB 595 :GOSUB 435:I=0:POSITION 7,116,127
90 REM "BOGY CONTROL"(妖怪的控制)
95 I=-(I+1)*(I<>NN)
100 XN=XPOS(I):YN=YPOS(I)
105 IF XN<12 OR XN>222 OR YN>188 THEN XN=RND(27)*8
    +12:YN=7:POSITION I,XN,YN:D=0:Q(I)=5:GOTO 140
110 XE=(XN-8)/8:YE=(YN-2)/8
115 G$=SCR$(XE,YE)
120 D=Q(I)
125 IF G$<>U$ THEN IF YL<>YE OR G$<>H$ THEN D
    =5:Q(I)=3-4*(X<XN):GOTO 135
130 IF D(I)=D THEN 150
135 XN=XE*8+12:YN=YE*8+7:POSITION I,XN,YN
140 DEF MOVE(I)=SPRITE(5,D,2,255+251*(D=5)):MOVE I:
    D(I)=D
145 REM "MEASURE COLLISION"(碰撞测试)
150 NI=CRASH(7):IF NI>-1 THEN IF ABS(YPOS(NI)-YPOS
    (7))<14 AND ABS(XPOS(NI)-XPOS(7))<14 THEN 320
```

```
155 IF F=1 THEN 380
160 REM "CONTROLLER"(操纵器)
165 S1=S:S=STICK(0):IF S>2 THEN S=3-(S>7)
170 X=XPOS(7):Y=YPOS(7)
175 XL=X/8-1:YL=(Y-2)/8
180 ON S GOTO 265,265,230,245
185 REM "MOVE SPRITE 7"(7#SP 运动)
190 V=0:K=0:CUT 7
195 IF SCR$(XL,YL)=" "THEN IF SCR$(XL,YL-1)<>H$
    THEN GOSUB 425:GOTO 750
200 XB=XL+(S1=1)-(S1=2):IF SCR$(XB,XL-1)=R$
    THEN GOSUB 430
205 IF STRIG(0)<>8 THEN 225
210 IF SCR$(XL-1,YL)=U$ THEN 370
215 IF SCR$(XL,YL)=U$ THEN 370
220 IF SCR$(XL+1,YL)=U$ THEN 370
225 GOTO 80
230 V=5:G$=SCR$(XL,YL)
235 IF G$<>H$ THEN IF G$<>U$ THEN S=0:V=0:GOTO
    365
240 GOTO 250
245 V=1:G$=SCR$(XL,YL-1)
250 IF G$<>H$ THEN S=0:GOTO 190
255 IF S1=0 OR K>0 THEN GOSUB 425:K=0
260 DEF MOVE(7)=SPRITE(0,V,1,4):MOVE 7:SPRITE S2:
    PAUSE 5:GOTO 80
265 V=4*S-1:K=1-(K+(K>3))*(S=S1)
270 G$=SCR$(XL+1+2*(S=2),YL):S$=SCR$(XL,YL)
275 XB=XL+1+2*(S=1):IF K>2 THEN IF SCR$(XB,YL-1)
    =R$ THEN GOSUB 430
280 IF SCR$(XL,YL-1)=R$ THEN XB=XL:GOSUB 430
```

```
285 IF G$<>U$ AND G$<>H$ THEN X=X+6*(S$<>"
    ")*((S=1)-(S=2)):GOTO 365
290 IF S$=" "THEN S=S1:GOTO 365
295 IF STRIG(0)=8 THEN IF G$=U$ OR S$=U$ OR SCR
    $(XB,YL)=U$ THEN 370
300 NI=CRASH(7):IF NI>-1 THEN IF ABS(XPOS(NI)-XPOS
    (7))<14 THEN 320
305 IF K<4 THEN DEF MOVE(7)=SPRITE(0,V,4-K,255):
    MOVE 7:SPRITE S2
310 GOTO 80
315 REM "THE END"(结束处理)
320 SPRITE S:SPRITE S1:SPRITE S2:ERA 0,1,2,3,4,5,6,7
325 FOR I=0 TO NN:POSITION I,0,0:D(I)=0:NEXT
330 LOCATE 8,9:PRINT "* * * * * * * * * * * * *"
335 LOCATE 8,10:PRINT "*      GAME OVER      *"
340 LOCATE 8,11:PRINT "* * * * * * * * * * * * *"
345 IF W>H THEN H=W:LOCATE 14,22:PRINT H;
350 IF STRIG(0)=1 THEN WW=WW-1:GOTO 70
355 GOTO 350
360 REM "JUMP SPRITE 7"(7#SP.跳)
365 G=2:Y=Y+3:GOTO 375
370 G=-9:X=X+(2*K+6)*((S=2)-(S=1)):Y=Y-16
375 F=1:SPRITE S,X,Y:ERA 7:SPRITE 3:POSITION 7,X,Y:S2=
    S:K=K*((S=2)-(S=1)):GOTO 80
380 NI=CRASH(7):IF NI>-1 THEN IF ABS(YPOS(NI)-Y)<12
    THEN 320
385 G=G+3+3*(G>6):X=X+K:Y=Y+G:SPRITE S,X,Y:PO-
    SITION 7,X,Y
390 XL=X/8-1:YL=(Y-2)/8:G$=SCR$(XL,YL):S$=SCR
    $(XL,YL-1)
395 IF G$=U$ THEN GOSUB 420:K=1:GOTO 260
```

```
400 IF S$=H$ THEN GOSUB 420:SPRITE 3,X,Y:SPRITE S2:S2
    =3:GOTO 80
405 IF G$=H$ THEN GOSUB 420:V=0:GOTO 260
410 GOTO 380
415 REM 子程序集
420 GOSUB 425:F=0:K=0:SPRITE S,X,Y:RETURN
425 CUT 7:X=XL*8+12:Y=YL*8+7:POSITION 7,X,Y:RE-
    TURN
430 LOCATE XB,YL-1:PRINT " ":R=R+1:W=W+15:LOCATE
    14,23:PRINT W;:PLAY "T1O3E1O4C":RETURN
435 GOSUB 510:I=0:F=0:K=0:C=0:R=0:LOCATE 10,21:
    PRINT "TIME"T:H=H+W:W=0:LOCATE 6,22:PRINT"HI
    -SCORE"H;:LOCATE 9,23:PRINT "SCORE"W;
440 RETURN
445 REM "ERROR HANDLING"(出错处理)
450 IF ERL=270 THEN XL=-27*(XL<1):GOSUB 425:MOVE
    7:RESUME 305
455 IF ERL=195 THEN XL=-27*(XL<0):GOSUB 425:S=S1:
    RESUME 265
460 IF YL>23 THEN RESUME 320
465 IF XL>27 THEN XL=0:GOSUB 425:RESUME
470 IF XL<0 THEN XL=27:GOSUB 425:RESUME
475 IF XB>27 THEN XB=0:MOVE 7:RESUME
480 IF XB<0 THEN XB=27:MOVE 7:RESUME
485 IF XL+1>27 THEN XL=0:GOSUB 425:RESUME
490 IF XL-1<0 THEN XL=27:GOSUB 425:RESUME
495 IF X<12 THEN X=228:RESUME
500 IF X>228 THEN X=12:RESUME
505 END
508 REM "BG"(背景)
510 CLS:SPRITE S:SPRITE S1:SPRITE S2:ERA 0,1,2,3,4,5,6,7:
```

```
    WW=WW+1:LOCATE 10,10:PRINT "STAGE "WW:GOSUB
    590:W$=" ":X$=" ":Z$=" ":RR=0:CLS:CGSET RND
    (2),RND(3)
515 FOR I=1 TO 10:W$=W$+U$
520 IF I>2 THEN X$=X$+U$
525 IF I>6 THEN Z$=Z$+U$
530 NEXT
535 FOR I=4 TO 18 STEP RND(2)+2:LOCATE RND(19),I:
    PRINT W$
540 LOCATE RND(21),I:PRINT X$:LOCATE RND(25),I:PRINT
    Z$:NEXT
545 FOR I=0 TO 27:LOCATE I,20:PRINT U$:NEXT
550 FOR I=0 TO 7:XX=RND(27):YY=RND(13)+3
555 FOR J=0 TO 4:LOCATE XX,YY+J:PRINT H$:NEXT:NEXT
560 FOR I=1 TO 10
565  XX=RND(27):YY=RND(19):IF SCR$(XX,YY)<>U$
     THEN 565
570 IF SCR$(XX,YY-1)=R$ THEN 565
575 LOCATE XX,YY-1:PRINT R$:NEXT
580 RETURN
585 REM "MUSIC"(音乐)
590 PLAY "T4Y2M0V9M1V9:T4Y1:T4"
595 PLAY "O3#G2#E#GO4C1:O3C2EGG1:O3E2CCR1"
600 PLAY "O3#G2#E#GO4C1:O3C2EGG1:O3E2CCR1"
605 PLAY "O3G2EO4CO3G1:C2EGO4C1:O2E2GGR1"
610 PLAY "O2#E2#D#C:E2DC:#E2#G#G"
615 RETURN
```

运行程序，一段音乐前奏过后，打印出如图 7—39 的游戏画面。(参考画面)，画面上有阶梯、悬崖、绳索和 10 个宝贝。玛丽必须躲过妖怪的追击、堵截，采完 10 个宝贝后过关。用 I$^{=}$操纵器的

方向键控制玛丽左、右运动和攀援绳索，按 A 键控制玛丽跳跃。若采完 10 个宝贝则奏乐，清屏，显示

STAGE 2

表示进入第二关。奏乐后，显示第二关的游戏画面。若第二关没有采完 10 个宝贝便被妖怪抓住或时间用完则转结束处理。屏幕显示如下画面

```
* * * * * * * * * * * * * *
*       GAME OVER         *
* * * * * * * * * * * * * *
```

按“START”键可继续游戏。

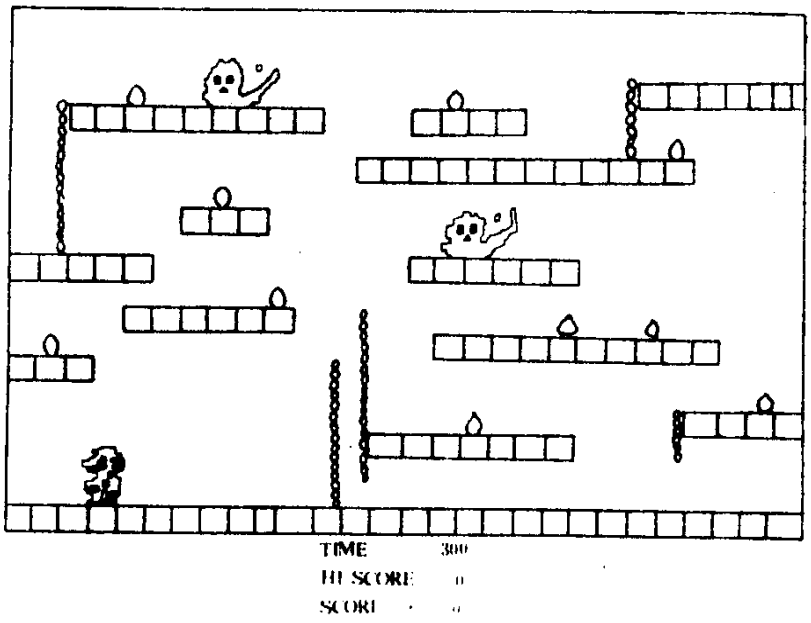


图 7—39 《玛丽采宝》游戏画面

程序说明：

1. 15～60 行为初始化程序。定义了 0# 显示 SP(玛丽跌)、1# 显示 SP(玛丽面向右跳)、2# 显示 SP(玛丽面向左跳)、3# 显示 SP(玛丽背面)。

2. 60～70 行游戏开始，调 435 行开始的子程序，打印游戏背景，时间赋值。

3. 75～85 行是游戏结束控制程序。若时间到则转结束处理；若玛丽采完 10 个宝贝，则转成功处理。

4. 90～140 行的作用是根据玛丽的座标，确定妖怪的运动方向，控制妖怪运动。

5. 145～155 行的作用是判断玛丽是否被妖怪抓住。

6. 160～310 行，用操纵器控制玛丽上、下、左、右运动。其中 205～260 行是按 A 键后的处理和玛丽执行上、下键的指令进行攀援绳索的运动。265～310 行是使玛丽执行左、右运动的指令，同时还考虑了按 A 键后的处理。

7. 315～355 行为结束处理程序。

8. 360～410 行为玛丽跳跃的处理程序。

9. 420 行子程序令 S^{π} 显示卡通显示。425 行子程序确定玛丽运动的起始座标。430 行子程序是对玛丽采到宝贝的处理。435 行子程序的作用是调背景绘制子程序绘制背景和打印时间、游戏者得分。

10. 445～500 行为出错处理程序。

11. 510～580 行是随机打印背景图案的子程序。其中 510 行完成有关变量的赋值；打印关次；随机设置背景和卡通的调色板号码。515～530 行完成对 W$、X$、Z$ 的赋值(这段程序也可以放在初始化程序里)。535～545 行打印悬崖、高山。550、555 行打印绳索。560～575 完成打印 10 个宝贝的操作。

程序 No. 7—39 中若删去 510～615 行和 90～140 行后，其结构与 No. 7—31 基本相同，因此读者可参考 No. 7--31 对 No. 7—39 程序进行分析理解，不多赘述。

No. 7—39 只能在 V3 版本中运行，若适用 BS. 2A 版本，需对出错处理语句和碰撞判断语句进行改造，读者可试改之。

第八章　实用程序的编写技巧

电子计算机的应用范围宏大广阔，已遍及人类作息的各个领域。本章将从游戏电脑的能力出发，撷沧海之一粟，分六个方面介绍常用实用程序的编程技巧，并通过实例重点介绍循环程序结构的优化。

8.1　计算求数类程序的编写技巧

8.1.1　马驮瓦问题——循环程序的优化

在介绍 F BASIC 语言的基本语句时，我们曾讨论过马驮瓦的问题。现仍以其为例，重点讨论循环程序的编程技巧。问题如下：

问题一：用 100 匹马驮 100 块瓦，大马驮 3 块、中马驮 2 块、两匹小马驮一块。求大、中、小马各用多少？（要求瓦要驮光，马要用了）。

1. 建立数学模型

设 A、B、C 分别代表大马、中马、小马的数量，根据题意可列出如下方程：

$$\begin{cases} A+B+C=100 & (1) \\ 6A+4B+C=200 & (2) \end{cases}$$

约束条件为 A≠0、B≠0、C≠0

2. 编制程序

根据方程(1)、(2)，我们很容易想到使用三重循环语句，用 A、B、C 的各种可能的值去试解，打印同时满足方程(1)、(2)的 A、B、

C，这就是问题的解。因此可写出程序 No. 8－1

```
10 REM "No. 8－1"
20 FOR A=1 TO 100
30 FOR B=1 TO 100
40 FOR C=1 TO 100
50 IF A+B+C<>100 THEN 80
60 IF 6*A+4*B+C<>200 THEN 80
70 PRINT A,B,C
80 NEXT:NEXT:NEXT
```

程序 No. 8－1 总的循环次数高达 100 万次（$100\times100\times100=10^6$），所以运行时间很长，高达 99 分钟。为缩短机时，必须对程序进行优化。

3. 分析问题

首先分析 A、B、C 三变量的循环终值是否都是 100。

在程序 No. 8－1 中没有考虑 A、B、C 不等于零的约束条件，故而循环终值均取 100。实际上三个变量的循环终值取 98 即可。读者可能因为终值减少 2 微不足道，但对多重循环来说，其作用是相当大的。我们可以计算一下：$98\times98\times98=941192$ 次，比 No. 8－1 减少近 6 万次的无效循环。

其次改写循环变量的终值。

由于循环变量的初、终值、步长可以是具体的数值，也可以是已赋值的变量或表达式。因此，根据方程(1)，我们可把三重循环终值改写如下：

```
20 FOR A=1 TO 98
30 FOR B=1 TO 99－A
40 FOR C=1 TO 100－A－B
```

这种形式与三重循环变量从 1 变化到 98 是等价的，但其作用却是巨大的。我们把程序 No. 8－1 改写为 No. 8－2。

```
10 REM "No. 8－2"
```

```
20 FOR A=1 TO 98
30 FOR B=1 TO 99-A
40 FOR C=1 TO 100-A-B
50 IF A+B+C=100 AND 6*A+4*B+C=200 THEN PRINT A,B,
   C
60 NEXT:NEXT:NEXT
```

运行 No. 8－2，15 分钟打印 6 组结果。可见，程序 No. 8－2 仅比 No. 8－1 调整了一下循环终值就使运行时间缩短了 84 分钟，速度提高了 85%。可以算出 No. 8－2 的循环次数为 161700 次，比 No. 8－1 省去了 838300 次的无效循环。

从上述事实可以看到多重循环的工作量是非常巨大的。同一程序由于循环终值的表达形式不同将对运行时间有着决定性的影响。从此可以看出循环程序优化的重要性。

通过程序 No. 8－2 可以得出结论：缩短初、终值之间的差距是减少循环次数的有效方法。但要注意这种减少不是随意的，必须由题意分析决定。要具体问题具体对待，不要盲目改写，以免造成运算错误或漏掉结果。

其实三个循环变量的终值也不可能都是 98。这是极为明显的。

根据题意若用 98 匹小马则驮 49 块瓦，再用大马、中马各一匹，虽然满足了方程(1)，用了 100 匹马，但只驮了 49＋3＋2＝54 块瓦，不能满足方程(2)，不符合 100 匹马驮 100 瓦的题意。同理，大马、中马也不可能用到 98 匹，否则，瓦将不够驮。

让我们进一步思索，采取有效措施减少循环次数。

4. 初、终值的深入探讨

改写方程(1)、(2)，可得

$$\begin{cases} B+C=100-A & (3) \\ 4B+C=200-6A & (4) \end{cases}$$

(3)式×6－(4)式得

$$5C+2B=400 \quad (5)$$

(4)式－(3)式得

$$5A+3B=100 \quad (6)$$

(3)式×4－(4)式得

$$3C-2A=200 \quad (7)$$

改写(5)、(6)、(7)三式则有

$$\begin{cases} B=(400-5C)/2 & (8) \\ A=(100-3B)/5 & (9) \\ C=(200+2A)/3 & (10) \end{cases}$$

表面上看原来的两个方程变成了三个方程，多了一个方程。但经过这样处理，却使问题简单多了。可以看出，每个方程只是两个变量之间的单一关系。

现在对(8)、(9)、(10)三个方程进行分析：由方程(8)，当 B 最少取 1 时，C＝79.6，取整数 C＝79；由(10)式，A 最小取 1 时，C＝67.3，取整数 C＝67。故可知 C 的变化范围：

$67 \leqslant C \leqslant 79$

根据方程(2)C 为奇数时不满足方程的解，所以 C 必为偶数。这样可知 C 的实际取值范围是：$68 \leqslant C \leqslant 78$ 且步长为 2

这个事实告诉我们，C 的初、终值不是 1 和 98，它的变化是遵循一定规律的。

同理，不难推出，A 的取值范围是从 2～17，步长为 3；B 的取值范围为 5～30，步长为 5，据此可以写出程序 No.8－3

```
10 REM "No.8－3"
20 FOR A=2 TO 17 STEP 3
30 FOR B=5 TO 30 STEP 5
40 FOR C=68 TO 78 STEP 2
50 IF A+B+C=100 AND A*6+1*B+C=200 THEN PRINT A,B,
   C
60 NEXT:NEXT:NEXT
```

读者不难算出，程序 No. 8－3 循环次数仅有 125 次。显然，本程序的运行速度是很快的。实测运行时间为 2.25 秒。

由此可见，同时增大初值和缩小终值并加大步长，会使程序的效率极大地提高。

5. 减少循环嵌套的重数

减少循环层次可进一步提高程序的运行速度。

根据(10)式，当 A 的值确定以后，C 的值也随之确定。因此，可去掉求 C 的循环，改为

C＝(200＋2＊A)/3。

同理，A、C 的值一旦确定，由(1)式可知

B＝100－A－C，

这又可省去一重循环。因此，循环次数仅为五次，它仅为程序 No. 8－1 的百万分之五。可见，程序的运行时间将会进一步缩短，见程序 No. 8－4。

```
10 REM "No. 8－4"
20 FOR A＝2 TO 17 STEP 3
30 C＝(200＋2＊A)/3
40 B＝100－B－C
50 IF A＋B＋C＝100 AND A＊6＋B＊4＋C＝200 THEN PRINTA,B,
   C
60 NEXT:END
```

程序 No. 8－4 的运行时间不到 1 秒钟。

经过上面的几步简化，程序 No. 8－4 好象是最简洁的了，但作为对循环程序结构的优化探讨，就本问题来说，循环体中的内容还可以斟酌。由于 A、B、C 三变量之间的关系完全由方程(8)、(9)、(10)所制约，并且在进行上述处理时，又已经考虑了约束条件，所以程序 No. 8－4 循环体内的判断语句完全可以省去。因而得到最终最简程序 No. 8－5

```
10 REM "No. 8－5"
```

```
20 FOR A=2 TO 17 STEP.3,C=(200+2*A)/3:B=100-A-C:
   PRINT A,B,C:NEXT
30 END
RUN          ↙    打印
2          30          68
5          25          70
8          20          72
11         15          74
14         10          76
17         5           78
```

程序 No. 8－5 不仅结构简单、程序短小、少占空间、节省内存，而且运行速度也是五个程序中最快的一个。

综上所述，我们围绕一组不定方程的求解过程，探讨了循环程序的结构优化。虽然从一个具体的实例出发，但引出的结论却具有普遍的意义。循环是程序的精华和核心，无论使用何种计算机语言编程几乎都离不开循环。因此，掌握循环程序的设计技巧对学习 F BASIC 来说无疑是极为重要的。

综合循环程序的编写技巧，可以归纳为以下五点：

①在步长不变的情况下，减少初、终值的差距是减少循环次数的有效方法。

②在初、终值一定的情况下，加大步长可以极大地提高程序的工作效率。

③减少循环嵌套的重数是提高程序运行速度的关键。

④调整和精简循环体内的语句是提高程序质量的有效措施。

⑤合理安排控制转向语句是及时脱离无效循环、避免重复计算的前提。

以上所有的这一切，又以尽量减少无用循环次数、努力避免大量无意义的重复工作为根本，这是高质量进行循环程序设计的核心。

以上为引申问题，得出结论，用了较大的篇幅探讨了循环程序设计技巧的各个方面，以后的叙述中，不可能都讨论得这样详尽，一般仅给出程序清单和附以简要说明。但只要根据以上介绍的方法步骤，读者不难看懂程序，也不难设计出高质量的循环程序。

8.1.2 勾股数组问题

问题二：我们的祖先很早就发现了“勾股定理”。这就是在直角三角形中，两直角边的平方和等于斜边的平方，写成数学表达式为：

$$A^2+B^2=C^2 \qquad (1)$$

满足上式的一组 A、B、C 的值叫做勾股数组。

要求编程求出给定值 C 以内的所有勾股数组。

关于求勾股数组的程序，已有许多文献介绍。但没有用整数 BASIC 语言编写的程序，本文给出用 F BASIC 语言编写的求勾股数组的程序。

解决该问题，仍然要用三重循环，对 A、B、C 的每一个值，从 1 到 C，逐个试验满足方程(1)的一组解。设给定值为 N，可编出程序 No. 8－6

```
10 REM “No. 8－6”
20 FOR A=1 TO N
30 FOR B=1 TO N
40 FOR C=1 TO N
50 IF A * A+B * B=C * C THEN PRINT A,B,C
60 NEXT:NEXT:NEXT
```

当 N＝100 时，可算出循环次数为 10^6 次。因此，程序 No. 8－6 的运行速度也是很慢的。

实际上，由方程(1)可知，A、B 的终值都不可能到 N，一旦确定了 A、B、C 中任两个数，第三个数就可根据方程(1)求出，从而去掉一重循环。但这样一来则要使用开方运算，若再调用开方子程

序，其运行速度将会更慢。因此，受F BASIC的运算能力制约(不能直接进行求根运算)，不能靠减少循环层次提高运算速度，从而只能在循环的初、终值上做文章。我们可把A的循环初值定为3，相应的B的初值应为A+1，则C的初值为B+1。这样一来将使循环次数减少50%多。另外，根据题意由方程(1)可知B的最小值为A+1，因而可求出A的循环终值为

$$A^2+(A+1)^2=N^2$$

$$A=\frac{N}{\sqrt{2}}\doteq 0.7N$$

这样以来，又可使循环次数减少30%。当A、B的值确定后，将仅有一个C值与其对应，因此应在C值确定后，及时退出C循环，这样又可使循环次数在现基础上减少43%。总之，经上述处理后，总的循环次数将由原来的P次变为

$$P(1-0.5)(1-0.3)(1-0.43)\doteq 0.2P$$

仅为程序No.8－6的五分之一。若取N＝100则求出所有结果仅需运行20万次。见程序No.8－7

```
5 REM "No.8－7"
10 CLS:CLEAR:INPUT "A*A+B*B=C*C,INPUT C=";N
20 FOR A=3 TO N:IF A*10>N*7 THEN END
30 FOR B=A+1 TO N－1
40 FOR C=B+1 TO N
50 D=A*A:E=B*B:F=C*C
60 IF F<D+E THEN 90
70 IF F>D+E THEN C=N:GOTO 90
80 PRINT A,B,C
90 NEXT:NEXT:NEXT
100 END
RUN        ↙ 显示
A*A+B*B=C*C,INPUT C=? □
```

键入 100 后，回车，则打印如下结果：

A^2	+	B^2	=	C^2
3		4		5
5		12		13
6		8		10
7		24		25
8		15		17
9		12		15
9		40		41
10		24		26
11		60		61
12		16		20
12		35		37
13		84		85
14		48		50
15		20		25
15		36		39
16		30		34
16		63		65
18		24		30
18		80		82
20		21		29
20		48		52
21		28		35
21		72		75
24		32		40
24		45		51
24		70		74
25		60		65
27		36		45
28		45		53

28	96	100
30	40	50
30	72	78
32	60	68
33	44	55
33	56	65
35	84	91
36	48	60
36	77	85
39	52	65
39	80	89
40	42	58
40	75	85
42	56	70
45	60	75
48	55	73
48	64	80
51	68	85
54	72	90
57	76	95
60	63	87
60	80	100
65	72	97

共 52 组勾股数。

8.1.3 一组求数程序

1. 求水仙花数

数学上把一个三位数等于其每位数的立方和的数称为水仙花数。如 $153=1^3+5^3+3^3$。

问题三：求 100～1000 之间的水仙花数。

设所求的数为 N，N 是一个三位数。再设 N 的百位、十位，个

位数分别为X、Y、Z，则有如下方程式：

$$X^3+Y^3+Z^3=N \qquad (1)$$

满足方程(1)的数N则为水仙花数。根据题意和F BASIC的除法运算特点，我们可以写出如下三个关系式：

百位数： X=N/100

十位数： Y=(N−X*100)/10

个位数： Z=N−X*100−Y*10

因此，可以安排一循环语句，对100～1000之间的数逐个判断，求出结果。见程序No.8−8

```
10 REM "No.8-8"
20 FOR N=100 TO 999
30 X=N/100:Y=(N-X*100)/10:Z=N-X*100-Y*10
40 IF N=X*X*X+Y*Y*Y+Z*Z*Z THEN PRINT X,Y,Z
50 NEXT
60 END
```

RUN　运行11秒，打印出4个水仙花数，分别为：

```
1  5  3
3  7  0
3  7  1
4  0  7
```

2. 问题四：试将100分成四个数，使得第一个数加4、第二个数减4、第三个数乘4、第四个数除以4后，所得的和、差、积、商都相等。

设四个数分别为A、B、C、D，根据题意可写出如下方程组：

$$\begin{cases} A+B+C+D=100 & (1) \\ A+4=B-4 & (2) \\ B-4=C*4 & (3) \\ C*4=D/4 & (4) \end{cases}$$

一般来说，对上述问题只要设置四重循环，就可求出答案。因此，可编写出程序No.8−9

```
10 REM "No. 8－9"
20 FOR A=1 TO 100
30 FOR B=1 TO 100
40 FOR C=1 TO 100
50 FOR D=1 TO 100
60 IF A+B+C+D<>100 THEN 100
70 IF A+4<>B－4 THEN 100
80 IF B－4<> C＊4 THEN 100
90 IF C＊4=D/4 THEN PRINT A,B,C,D: END
100 NEXT:NEXT:NEXT:NEXT
```

显然，No. 8－9 运行时间太长，效率太低。分析方程组，可改写如下：

$$\begin{cases} A+B+C+D=100 \\ A+4=B-4 \\ A+4=C*4 \\ A+4=D/4 \end{cases}$$

化简后有

$$\begin{cases} A+B+C+D=100 & (1)' \\ B=A+8 & (2)' \\ C=A/4+1 & (3)' \\ D=4*A+16 & (4)' \end{cases}$$

由方程(2)'、(3)'、(4)'可见，只要 A 值确定，其它三值都可确定。所以，只要对 A 循环就可求出答案。从而编写出程序 No. 8－10

```
10 REM "No. 8－10"
20 FOR A=1 TO 97
30 B=A+8:C=A/4+1:D=4＊A+16
40 IF A+B+C+D=100 THEN PRINT A,B,C,D
50 NEXT
RUN  ↙   立即打印结果为
12    20    4    64
```

程序 No. 8－10 比 No. 8－9 的循环次数减少了 100 万倍。

由方程式(3)'可知，A 是 4 的整数倍。因此，A 的循环步长可以加大。这样，程序的运行速度将进一步提高。另外，由

$$B+C+D=5A+(1/4)A+25$$

可知 A 的最大取值不会大于四数之和的六分之一。所以，No. 8－10 中 A 的循环终值可由 97 缩小为 100/6＝16。

现在编写一个适用 32767 之内数的通用程序，见 No. 8－11。

```
10 REM "No. 8－11"
20 CLS:CLEAR:INPUT "A+B+C+D=N, INPUT N=";N
30 FOR A=4 TO N/6 STEP 4
40 B=A+8:C=A/4+1:D=A*4+16
50 IF A+B+C+D=N THEN PRINT A,B,C,D:GOTO 70
60 NEXT
70 A$=INKEY$
80 GOTO 20
```

RUN ↙ 显示

A+B+C+D=N,INPUT N=？□，键入 100 则立即打印

12	20	4	64	键入 200 则有
28	36	8	128	键入 300 则有
44	52	12	192	键入 400 则有
60	68	16	256	键入 500 则有
76	84	20	320	键入 10000 则有
1596	1604	400	6400	

既使键入 30000，也仅用 2 秒钟即可显示结果。

4796　4804　1200　19200

可见，程序 No. 8－11 的运算速度是较高的。

3. 已知四位数 3025 有一个特殊的性质，它的前两位数(30)与后两位数(25)的和的平方就是自己本身。即$(30+25)^2=3025$。

问题五：试编程求出所有具有上述性质的四位数。

根据题意编出程序 No. 8－12

```
10 REM "No. 8－12"
20 FOR N=1000 TO 9999
30 F=N/100:L=N－F * 100
40 P=(F+L) * (F+L)
50 IF P=N THEN PRINT F+L"X"F+L"="N
60 NEXT
70 END
RUN    ↙    打印
45×45=2025
55×55=3025
```

继续运行后，打印

```
OV ERROR IN 40
```

出现溢出错误而停机。键入

```
PRINT N, F+L 查看变量情况，发现
N=9587    F+L=182
```

而 182 的平方是 33124，已超过 F BASIC 的数值范围。

其实我们求的结果都是 4 位数，F＋L 的值不会超过 99（若超过 99 则为 5 位数，不合题意）。因此，必须在程序中加入限制语句，以保证程序正常运行，见程序 No. 8－13。

```
10 REM "8－13"
20 FOR N=1000 TO 9999
30 F=N/100:L=N－100 * F:P=F+L
40 IF P>100 THEN 60
50 M=P * P:IF M=N THEN PRINT P" * " P "=" N
60 NEXT
70 END
RUN    ↙    运行 80 秒，打印出 3 个结果：
45×45=2025
55×55=3025
```

99×99=9801

4. 问题六：求素数

所谓素数就是除了能被 1 和它本身整除外，不能被其它任何整数整除的自然数。因此，判定一个自然数 X 是否为素数，可以将 X 被 2～X/2 各整数相除，若都不能整除，则 X 是素数。因此可编写程序 No. 8－14 进行判别。

```
10 REM No. 8－14
20 CLS:CLEAR:INPUT "N="; N
30 FOR X=3 TO N
40 FOR I=2 TO X/2
50 IF X MOD I=0 THEN I=X/2:NEXT
60 NEXT
70 PRINT X;" ";
80 NEXT:END
RUN    ↙    输入 50 则打印
3  5  7  11  13  15  17  19  23  29  31  37  41  43  47
```

并最后打印

```
NF ERROR IN 80
```

表示 80 行多了一个 NEXT。解决的方法是在 30 行对 X=N 进行拦截处理。另外，由于素数不会是偶数，所以 X 的步长可改为 2，跳过偶数。这样可使运算速度提高一倍。修改后的程序见 No. 8－15

```
10 REM No. 8－15
20 CLS:INPUT "N="; N
30 FOR X=3 TO N+2 STEP 2:IF X>N THEN END
40 FOR I=2 TO X/2:IF X MOD I=0 THEN I=X/2:NEXT
50 NEXT:PRINT X " ";:NEXT
RUN    ↙    键入 100 ↙ 则打印
3  5  7  11  13  17  19  23  29  31  37  41  43  47  53  59
61  67  71  73  79  83  89  97
```

5. 对于一个正整数 A，若 A 的高低位上的数值对称相等，则

称 A 为回文数。如 121、232、1234321 等。

问题七：求 32000 以内的完全平方的回文数。

由回文数的性质可知，回文数分别为 3、5、7……位数，对于问题七要求的回文数则只能是 3 位数或 5 位数。因此，我们可以设置一层循环，循环变量的初值为 11(因为 11 的平方幂为 121 是三位数中最小的平方回文数)，终值为 180(因为 180 的平方为 32400，已超过问题要求的取数范围)，然后在循环体内设置判断回文数的语句，对每个循环变量的平方幂进行判别，若是回文数则把它打印出来，若不是则循环变量加 1，直到循环结束，则可求出问题七要求的全部回文数。编程的关键是解决回文数的判别问题。根据回文数的特点，对给定的数值 A，我们把 A 的高低位数字全部对调后的数称为 A'，若有 A＝A'则可判定 A 为回文数。根据上述思路，编程如 No. 8－16

```
10 REM No. 8－16
20 FOR I=11 TO 180
30 N=I*I:A$=STR$(N):B$=RIGHT(A$,3)
40 IF LEFT$(A$,1)=" " THEN A$=RIGHT$(A$,LEN(A$)
   －1)
50 C$=MID$(A$,3,1)+MID$(A$,2,1)+MID$(A$,1,1)
60 IF B$=C$ THEN PRINT N"=" I "*" I
70 NEXT
80 END
RUN     ↙     打印六个结果
121=11*11
484=22*22
676=26*26
10201=101*101
12321=111*111
14641=121*121
```

程序 No. 8－16 中的 40 行是为了消去数字转换为字符串后前面出现的空格。50 行的作用是颠倒数值的高、低位。由于问题所求的回文数最多为五位数，所以仅比较三位数就可判定回文数。如“14641”，其后三位为“641”，通过 30 行赋给 B$，其前三位“146”颠倒后也是“641”，则由 50 行赋给 C$，经 60 行判断 B$＝C$，所以“14641”是回文数。再如 130 的平方幂为“16900”它的后 3 位 B$＝“900”，而 C$＝“961”，B$≠C$ 所以“16900”不是回文数。

6. 预测牛数

问题八：买来一头母牛，从第一年起，每年生一头小母牛，每头小母牛从第四个年头起，每年也生一头小母牛。问在第二十年时，牛的总数有多少？

设 N1 表示当年出生的小母牛数，N2、N3 分别表示生长到第二年、第三年的小母牛数，N4 表示第四年和四年以上的母牛数。那么，对于某一年的 N4 应该是前一年的 N4 与 N3 之和，而 N3 则是前一年的 N2，N2 是前一年的 N1，而 N1 又等于当年的 N4。所以，通过辗转赋值和套入循环可编写出程序 No. 8－17

```
10 REM No. 8－17
20 N4＝1:N1＝1:N2＝0:N3＝0
30 FOR I＝2 TO 20
40 N4＝N4＋N3:N3＝N2:N2＝N1:N1＝N4
50 IF I<4 THEN 70
60 N＝N1＋N2＋N3＋N4:PRINT I, N
70 NEXT
80 END
```

程序中 50 行的作用是当小于四年时，只有买来的母牛生小牛，新生的小母牛还不能生小牛，所以只辗转赋值，不求和。只有到第四个年头，第一年生的小母牛开始生小牛，所以开始求和，这时 50 行不再起作用。

运行程序，可以第 4 年开始打印一直到第二十年的总牛数。分

别如下：

4	6
5	9
6	13
7	19
8	28
9	41
10	60
11	88
12	129
13	189
14	277
15	406
16	595
17	872
18	1278
19	1873
20	2745

从运行结果可以看出，某年的牛数等于其前一年的牛数和前三年的牛数之和。如第15年的牛数是406头，其前一年的牛数是277头，其前三年的牛数是129头，则有277+129=406，因此，程序No. 8－17还可以优化。

由题意，我们知道头三年的牛数分别为2、3、4头，如果我们用K、L、M分别表示前三年、前两年、前一年每年的牛数，则它们的初值分别为K=2. L=3，M=4。因而可写出程序No. 8－18

```
10 REM No. 8－18
20 K=2:L=3:M=4
30 FOR I=4 TO 20
40 N=M+K:PRINT I,N
50 K=L:L=M:M=N
```

```
60 NEXT
```

运行程序，仍然打印上述结果。

对问题八，No. 8－18 还可以进一步优化。优化的方法是用下标变量取代简单变量的辗转赋值。程序如下：

```
10 REM No. 8－19
20 DIM A(20):A(1)=2:A(2)=3:A(3)=4
30 FOR I=4 TO 20:A(I)=A(I－1)+A(I－3):PRINT I,A(I):NEXT
```

程序 No. 8－19 应是解决问题八的最简程序了。

8.2 逻辑推理类程序的编写技巧

电脑不仅具有较高的数学运算速度，而且还具有极强的逻辑判断能力，可以迅速准确地进行逻辑推理，作出果断正确的判断。计算机进行逻辑判断常用的方法之一就是穷举法。所谓穷举法就是列出各种事件发生的可能，根据提供条件进行逐个验证。若符合所有条件则判为真，否则判为假。下面通过几个具体事例介绍电脑用于逻辑推理问题时的编程方法。

8.2.1 到底谁说谎

问题九：A、B、C 三人在一起争论，A 指责 B 正在说谎；B 指责 C 正在说谎；而 C 又指责 A 和 B 都在说谎，试编一程序判断到底谁在说谎。

对于这类问题，由于只有说谎和不说谎两种情况，所以可作一个逻辑上的假定：说谎设为“1”，说实话设为“0”。对每一个人来说，都有说谎和讲真话两种可能，那么对于三个人（三个变量）就有八种情况，分别列于表 8－1。

表 8－1

A	B	C	A	B	C
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	1	1	0
0	1	1	1	1	1

对于 A 指责 B 说谎，也有两种情况：若 A 说真话则 B 说假说；反之，若 A 说假话则 B 是老实人。即有

A＝0　　B＝1

A＝1　　B＝0

这种关系可以用关系式

ABS(A－B)＝1

表示，意思是不管 A、B 哪一个变量是 1，关系式的值恒为 1。同理，B 指责 C 说谎，可以表示为：

ABS(B－C)＝1

C 指责 A 和 B 都在说谎则可表示为：

ABS(C－A＊B)＝1

有了以上三个关系式就可以安排程序了。由于 A、B、C 都有说谎话和说实话两种可能、八种情况。所以，我们很容易想到用三重循环来解决，而满足题意的三个条件则使用条件语句来判断。为使程序简洁，我们采用逻辑表达式来表示这三个条件。编程如下：

```
5 REM No. 8－19
10 FOR A＝0 TO 1
20 FOR B＝0 TO 1
30 FOR C＝0 TO 1
40 IF ABS(A－B)＝0 OR ABS(B－C)＝0 OR ABS(C－A＊B)＝0
   THEN 80
50 IF A＝1 THEN PRINT "A＝1"
60 IF B＝1 THEN PRINT "B＝1"
70 IF C＝1 THEN PRINT "C＝1"
```

```
  80 NEXT
  90 NEXT
  100 NEXT
110 END
RUN    ↙    打印
A=1
C=1
```

可见，真正的说谎者是 A 和 C。

程序 No. 8－19 证实了用计算机进行逻辑推理判断是行之有效的。这表明计算机具有一定的逻辑思维能力，也许这就是把计算机称为“电脑”的原因。人类生活中的许多问题往往千丝万缕、错综复杂，有着多个变量、多个制约关系。人的智力不可能一下子涉及到问题的全部细节，常常难于迅速、准确地做出判断，而计算机则正是解决这类繁琐的脑力劳动的有效工具。

8.2.2 谁是罪犯

问题十：有 X1、X2、X3、X4、X5 五个人可能参与了一起凶杀案。关于哪些人参与此案虽没有直接的证据却有以下几条可靠的线索：

(1) 若 X1 参与此案则 X2 也参与此案；

(2) X4、X5 中至少有一人参予此案；

(3) X2、X3 中只有一人参与作案；

(4) X3 和 X4 要么同时参与作案，要么都与本案无关；

(5) 如果 X5 参与作案，那么 X1、X4 也一定参与此案。

由于本问题只有参与作案与不参与两种情况，所以我们设参与作案为“1”，不参与为“0”，根据以上五条线索可列出如下五个关系式：

- 根据线索(1)　可列出　X1＝X2
- 根据线索(2)　可列出　X4＋X5＞0

- 根据线索(3)　可列出　X2≠X3
- 根据线索(4)　可列出　X3＊X4＋(1－X3)＊(1－X4)＝1
- 根据线索(5)　可列出　X1＊X4＝X5

由于X1～X5五个人都有可能参与作案，所以，我们引入五重循环编程。见No. 8－20

```
5 REM No. 8－20
10 FOR X1=0 TO 1
20 FOR X2=0 TO 1
30 FOR X3=0 TO 1
40 FOR X4=0 TO 1
50 FOR X5=0 TO 1
60 IF X1<>X2 THEN 130
70 IF X4+X5<=0 THEN 130
80 IF X2=X3 THEN 130
90 IF X3＊X4+(1－X3)＊(1－X4)<>1 THEN 130
100 IF X4＊X1<>X5 THEN 130
110 PRINT "X1 X2 X3 X4 X5"
120 PRINT X1" "X2" "X3" "X4" "X5
130 NEXT:NEXT:NEXT:NEXT:NEXT
140 END
RUN　　↙　　打印
X1  X2  X3  X4  X5
0   0   1   1   0
```

因此判定X3、X4参与了凶杀案。

8.2.3　谁去执行任务

问题十一：某侦察排长接到一项紧急任务，要他在代号为A、B、C、D、E、F六个人中挑选若干人去完成一项艰巨的侦察任务，人选的配备要求，必须注意以下六点：

(1) A、B两人中至少去一个人。

(2) A、D不能同去。

(3) A、F、E 三人中要派两人去。

(4) B、C 两人都去或都不去。

(5) C、D 两人中去一人。

(6) 若 D 不去，则 E 也不去。

试编程判断应该让谁去，去几人。

该问题同问题十大同小异，我们设去执行任务的为“1”，不去的为“0”，则可列出如下六个关系式：

$A+B>0$

$A\neq D$

$AE+AF+EF=1$

$BC+(1-B)(1-C)=1$

$C+D=1$

$D-E=0$

根据以上六个关系式，引入六重循环，编程如下：

```
5 REM No. 8-21
10 FOR A=0 TO 1
20 FOR B=0 TO 1
30 FOR C=0 TO 1
40 FOR D=0 TO 1
50 FOR E=0 TO 1
60 FOR F=0 TO 1
70 IF A+B<=0 THEN 150
80 IF A=D THEN 150
90 IF A*E+A*F+E*F<>1 THEN 150
100 IF C+D<>1 THEN 150
110 IF B*C+(1-B)*(1-C)<>1 THEN 150
120 IF D-E<> 0 THEN 150
130 PRINT "A B C D E F"
140 PRINT A"  "B"  "C"  "D"  "E"  "F
150 NEXT:NEXT:NEXT:NEXT:NEXT:NEXT
```

```
RUN          ↙      则打印
A  B  C  D  E  F
1  1  1  0  0  1
```

结果表明应选A、B、C、F四人去执行任务。

8.2.4 哪枚是假币

问题十二：八枚硬币问题。

有八枚硬币A、B、C、D、E、F、G、H，已知其中的一枚是假币，假币的重量与真币不同。要求用最少的次数找出假币，并判别它较真币为轻还是重。这就是著名的“八枚硬币问题”。

解决八枚硬币问题的算法有多种，我们的算法思路是：假定有一等臂天平，我们从八枚硬币中任意取出六枚，比如取A、B、C、D、E、F；然后将这六枚硬币分为两组，每组三枚，比如A、B、C一组、D、E、F为一组；把两组硬币分别放在天平的两臂上进行称量比较；若A+B+C<D+E+F，则我们得知假币就在这六枚之中，而与G、H无关；接下来天平两端各去掉一个硬币，比如去掉C和F；然后天平两边再调换一枚硬币，比如B、D对调；再进行第二次称量，如仍有A+D<B+E，则表示对换B、D后不等情况没有改变，由此说明去掉的C和F是真币，也说明B和D是真币（若C、F中有假币则天平要反向倾斜，即有A+D>B+E；若有A+D=B+E，则说明假币在C、F中），至此，可判定假币在A、E中；第三次拿A与一枚真币（如B）进行比较，若A=B则E为重币，否则A为轻币。经过三次比较找出了假币，并且确定了假币是重还是轻。显然三次比较是最少的。

根据上述算法思路，编写程序如No. 8－22。

```
5 REM No. 8－22
10 CLS:PRINT " A,B,C,D,E,F,G,H":INPUT A,B,C,D,E,F,G,H
20 S1=A+B+C:S2=D+E+F
30 IF S1>S2 THEN 90
40 IF S1<S2 THEN SWAP A,D:SWAP B,E:SWAP C,F:GOTO 90
```

```
50 IF G=H THEN PRINT "EQUAL!":END
60 IF G>H THEN 80
70 SWAP G,H
80 X=G:Y=H:Z=A:GOTO 140
90 IF A+D>B+E THEN 120
100 IF A+D<B+E THEN 130
110 X=C:Y=F:Z=A:GOTO 140
120 X=A:Y=E:Z=B:GOTO 140
130 X=B:Y=D:Z=A:GOTO 140
140 IF X>Z THEN PRINT X" is heauy!":END
150 PRINT Y " is light!":END
RUN    ↙    屏幕上显示
A,B,C,D,E,F,G,H
?□
```

闪烁的光标等待我们输入八枚硬币的重量，如果我们选 6,6,6,5,6,6,6,6，回车后，立即打印"5 is light !"。在屏幕上的综合显示为：

```
A,B,C,D,E,F,G,H
? 6  6  6  5  6  6  6  6
   5 is light!
```

表示重量为 5 的是轻币。

```
RUN    ↙
A,B,C,D,E,F,G,H
? 7, 7, 7, 7, 7, 7, 7, 7
   EQUAL!
   OK
```

表示八枚硬币重量相等。

```
RUN    ↙
A,B,C,D,E,F,G,H
? 5, 5, 7, 5, 5, 5, 5, 5
7 is heavy!
```

OK

程序说明：

1. 30～80 行为第一次比较。其中 30、40 行是 A+B+C≠D+E+F 处理。50～80 行是相等的处理，当 A+B+C=D+E+F 时，假币在 G、H 两枚中。如果 G=H 则表示八枚硬币相等；如果 G>H 则转 80 行，把重的赋给 X、轻的赋给 Y，真币的重量赋给 Z 后转 140 行进行第三次比较。140、150 两行是第三次比较程序，如果 X 比真币重则判定 X 为重币；若 X 与真币相等则判定 Y 为轻币。

2. 90～130 行为第二次比较程序。其中 110、130 行是针对 A+D>B+E、A+D<B+E、A+D=B+E 三种情况的分别处理，目的是对照三种情况对 X、Y、Z 赋以不同的值以便于进行第三次比较。

3. 140～150 行是各种情况下第三次比较的共同比较程序。

另外，40 行中的 SWAP 语句是为了与 30 行共用 90 行的比较程序。70 行的 SWAP 语句是为了与 60 行共用 80 行的赋值语句。这样可使程序简化。

8.2.5 戴维住哪间房

精明的旅客住丽云旅馆时总是予订单号房间，这是因为单号房间都在旅馆大楼的东部，从那里可以俯瞰美丽的海滨。而双号房间在大楼的另一边，从那里望见的却是屠宰场。因此，戴维为他和他的朋友来这里度假，预订的是九个相连的单号房间。

第二天早上，朋友们来到了旅馆，每人选了一个房间，放进了各自的行李后就出去游览了。晚上回到旅馆，朋友们都有点醉了。九把钥匙都集中在戴维手里，谁也记不起自己的房间号码，而且谁也不愿意让别人开自己的房间。弄得戴维一时不知所措。

问题十三：请根据下面提供的六条线索，替戴维确定每个朋友的房间号码。

(1)A、F 和 I 记得他们各在两个相连质数号房间的后面一间。

(2)B 记得他的房间号数比 I 的大 4。

(3)C 的号数比 H 的大。他到 H 房间的距离是到 F 房间的一倍(这里的距离是指相隔的房间间数)。

(4)G 的号数是九个人中唯一的两位数,它和 E 的号数相差 4。

(5)E 和 I 不是邻居。

(6)F 的号数比 A 小。

根据提供的六条线索,分析如下:

1. 根据线索 4 可以判定 G 的房间号数是 99。理由是,若 G 是 11 的话,11 的前面只有 5 个奇数,房间不够 8 间。

2. 判定 G=99 后,可知戴维订的房间号码为 99～115,见下表

99	101	103	105	107	109	111	113	115
	*	*		*	*	*	*	

表中带星号的号码表示该数是质数。

3. 由线索 4 和 G 等于 99 可知 E=103。

4. 根据线索 1,在两个相连质数后面的号码只有 105、111、113、115 四个房间,所以可知 A>103、F>103、I>103,并且有:A、F、I 都不等于 107、109。

5. 由线索 5 可知 I>109

综上所述,再结合线索 2、3、6 可写出下列关系式:

A>103、A≠107、A≠109

F>103、F≠107、F≠109

I>109

B−J=4

C>H

C−H=2|C−F|

F<A

G=99、E=103

在上述关系式中,除 G、E 的房间号码确定外,其它 7 人都有

七种可能。因此，若仿照 No. 8－20～No. 8－22 的形式编程，要使用 7 重循环，每个循环要进行 7 次，总循环次数为 $7^7 \doteq 82$ 万次，程序执行速度太慢。我们采取辗转交换变量内容的方法，仅使用一层循环就可求出答案。程序如下：

```
5 REM No. 8－23
10 CLS:G=99:E=103:A=101:B=105:C=107:D=109:F=111:H=
   113:I=115
20 FOR J=1 TO 9
30 IF A<105 OR A=107 OR A=109 THEN SWAP A,B
40 IF B－I<> 4 THEN SWAP B,C
50 IF C<=H THEN SWAP C,D
60 IF C－H<> 2 * ABS(C－F) THEN SWAP D,F
70 IF F<105 OR F=107 OR F=109 THEN SWAP F,H
80 IF I<111 THEN SWAP H,I
90 IF F>A THEN SWAP A,F
100 IF B－I<>4 THEN SWAP B,I
110 NEXT
120 IF A>103 AND A<>107 AND A<>109 AND F>103 AND F<
   >107 AND F<>109 AND I>109 AND B－I=4 AND C>H
   AND C－H=2 * ABS(C－F) AND F<A THEN 140
130 GOTO 20
140 PRINT "A="A," B="B," C="C," D="D," E="E," F="F,"
   G="G," H="H," I="I
150 END
RUN      ↙      立即打印出
A=113            B=115
C=109            D=107
E=103            F=105
G=99             H=101
I=111
```

把结果与六个线索相对照检查，结果完全正确。

程序说明：

1. 10 行为初始赋值程序。其中除 G、E 为已知，赋以相应数值外，其它七个变量可任意赋值，目的是把数据输入程序。

2. 30～100 行为本程序的关键，从 30 行开始依次交换 A、B，B、C，C、D，D、F，F、H 和 H、I 及 A、F。30～90 行的交换是依据所列关系式是否满足为前提的。为避免变量交换进入死循环，特加入 100 行以接通交换链，使交换顺序 A→B→C→D→F→H→I→B。

3. 由 120 行判断是否符合解题条件，若满足条件则转 140 行输出结果；若不满足条件则经 130 行转 20 行进行下一轮交换循环。实际上，问题十三仅循环八次就得出了结果。

以上五个程序从不同的侧面说明了逻辑推理类程序的编写技巧，读者可举一反三。

8.3 绘图类程序的编写技巧

F BASIC 有着极强的键控绘图能力，在 BG GRAPHIC 状态下，用户可以任意挥洒、纵横驰骋，发挥丰富的想象力，绘制出各种精彩的图案，且不占用户内存，这是任何 BASIC 语言都难以比拟的。但是，键控绘图也有它的缺陷，即屏幕的利用效率低。我们知道，背景显示面有 24 行、28 列，而键控绘图只能使用 21 行，下面的三行用于绘图提示而不能使用，从而使绘图的有效面积缩小了 12%。编程绘图则可以使用整个背景面，这就是它优于键控绘图的地方。

在键盘的说明书中仅介绍了键控绘图，没有介绍编程绘图的方法，同时 F BASIC 语言中也没有诸如 PLOT、HLINE、VLINE 之类的绘图指令，从而使许多朋友产生误解，认为 F BASIC 不能编程绘图。其实，只要充分利用光标控制的有关语句，使用较小的

程序量也可以绘出较精彩的图案。第五章曾介绍了绘制简单图案的程序，现在进一步讨论简单图形的绘制编程及图形的变化技巧。

8.3.1 简单图形的绘制

例 8.1 编程绘制用方框围起来的米字形图案。

```
5 REM No. 8－24
10 CLS
20 FOR X＝0 TO 25
30 LOCATE X,0:PRINT "＊"
40 LOCATE X,21:PRINT "＊"
50 LOCATE X,10:PRINT "＊"
60 LOCATE X,(21－21＊X/25):PRINT "＊"
70 LOCATE X,21＊X/25:PRINT "＊"
80 IF X＜22 THEN LOCATE 13,X:PRINT "＊":LOCATE 0,X:
   PRINT "＊":LOCATE25,X:PRINT "＊"
90 NEXT
100 END
RUN    ↙  绘出如图 8－1 的图案
```

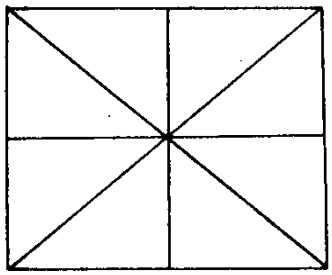


图 8－1　米字形图案

例 8.2 编程绘制长方体图案

```
5 REM No. 8－25
10 CLS:A$＝CHR$(199):B$＝"＊":C$＝"@"
20 FOR X＝0 TO 9
```

```
30 LOCATE 0,X:PRINT B$
40 LOCATE X,0:PRINT B$
50 LOCATE 9,X:PRINT B$
60 LOCATE X,9:PRINT B$
70 LOCATE 17+X,9:PRINT C$
80 LOCATE 17+X,18:PRINT C$
90 LOCATE 17,9+X:PRINT C$
100 LOCATE 26,9+X:PRINT C$
110 NEXT
120 FOR X=1 TO 16
130 LOCATE X,9*X/17:PRINT A$
140 LOCATE X,9*X/17+9:PRINT A$
150 LOCATE X+9,9*(X+9)/17-5:PRINT A$
160 IF X<>8 THEN LOCATE X+9,9*(X+9)/17+4:PRINT A$
170 NEXT
180 END
RUN    ↙  打印长方体图案，见图 8-2。
```

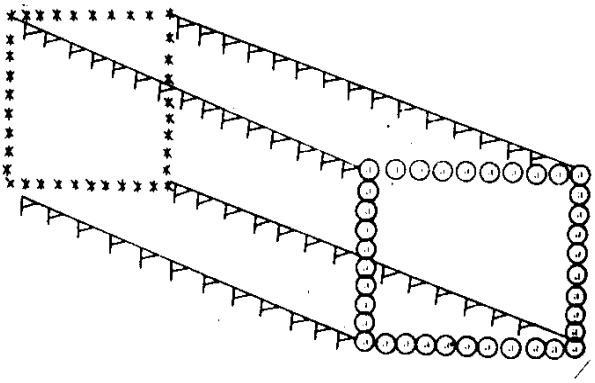


图 8-2　长方体

例 8.3　编程绘制五个正方形套在一起的图案。

```
5 REM No. 8-26
10 CLS:X=13:Y=11:A$=CHR$(253)
20 FOR N=1 TO 10 STEP 2
```

```
30 FOR L=-N TO N
40 LOCATE X+L,Y-N:PRINT A$
50 LOCATE X+N,Y+L:PRINT A$
60 LOCATE X-L,Y+N:PRINT A$
70 LOCATE X-N,Y-L:PRINT A$
80 NEXT:NEXT
90 END
```

RUN ↙ 则打印如图 8－3 的五个正方形套在一起的图案。图案从中心最小的正方形开始，以顺时针方针旋转打印。

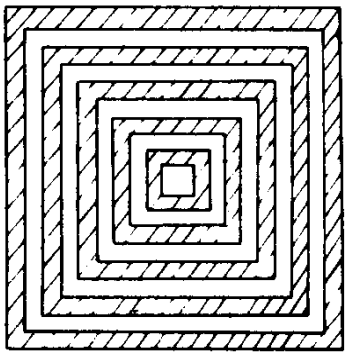


图 8－3 五套环

程序 No. 8－26 的技巧在于使用两重循环从而使程序得到极大的简化。

例 8. 4 编程使图 8－3 的图形，顺时针从内向外绘制，绘完后，再令其逆时针从外向内消失，消失后再绘制，周而复始。

欲达到这一要求，我们只要令程序 No. 8－26 中的循环体反向循环打印空格，即可使已打印的图形顺序消失。见程序 No. 8－27。

```
5 REM No. 8-27
10 CLS: X=13:Y=11:A$=CHR$(253):B$=" "
20 FOR N=1 TO 9 STEP 2
30 FOR L=-N TO N
```

```
40 GOSUB 120
50 NEXT:NEXT
60 SWAP A$,B$
70 FOR N=9 TO 1 STEP -2
80 FOR L=N TO -N STEP -1
90 GOSUB 120
100 NEXT:NEXT
110 SWAP A$,B$:GOTO 20
120 LOCATE X+L,Y-N:PRINT A$
130 LOCATE X+N,Y+L:PRINT A$
140 LOCATE X-L,Y+N:PRINT A$
150 LOCATE X-N,Y-L:PRINT A$
160 RETURN
RUN ↙
```

屏幕上的五套环图形反复显示、消失，产生了很强的动感。

程序中 120～160 行为绘图子程序；20～50 行为绘图定位程序；70～100 行为擦图定位程序；60、110 两行实现绘图、擦图的转换。

若使擦图也按顺时针方向从外向内擦，可把 80 行改为

```
80 FOR L=-N TO N
```

则可使绘图、擦图全部以顺时针方向进行。

若使擦图过程与绘图过程一样，都从内向外顺时针进行，可删去 60～100 行。那样，程序将更为简洁。见程序 No. 8－28

```
5 REM No. 8-28
10 CLS:X=13:Y=11:A$=CHR$(253):B4=" "
20 FOR N=1 TO 9 STEP 2
30 FOR L=-N TO N
40 LOCATE X+L,Y-N:PRINT A$
50 LOCATE X+N,Y+L:PRINT A$
60 LOCATE X-L,Y+N:PRINT A$
```

```
70 LOCATE X-N,Y-L:PRINT A$
80 NEXT:NEXT
90 SWAP A$,B$:GOTO 20
100 END
```

8.3.2 简单图形的变形

看到下面的图形(见图 8－4),读者肯定会说这是用键控绘图画出来的。其实这是编程画出来的。听到是编程绘图,不少朋友会以为程序量肯定不算小。其实并不然,图 8－4 仅是对程序 No. 8－26 稍加改造后绘出的,这种改造称为图形的变形。现在我们就介绍改造的方法。

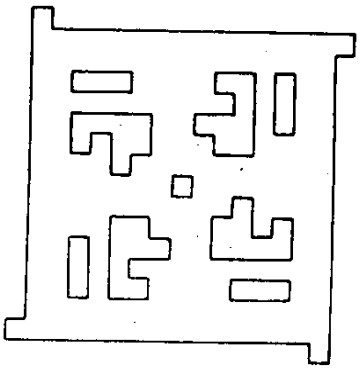


图 8－4

1. 若把程序 No. 8－26 中的 30 行改为

```
30 FOR L=-N-2 TO N+2
```

则可绘出精美的花窗图案,见图 8－5。

2. 若把程序 No. 8－26 中的 30 行改为

```
30 FOR L=-N+1 TO N+2
```

可绘出五风车嵌套图案,见图 8－6。

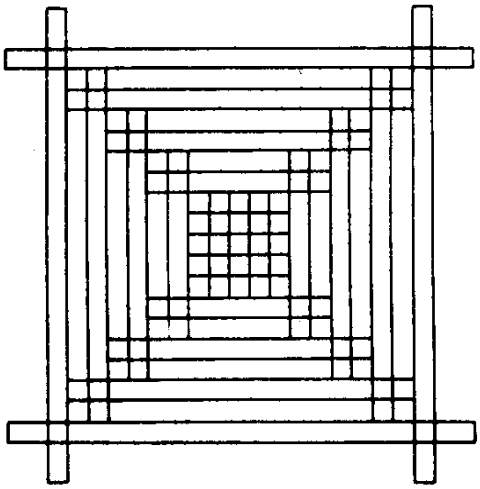


图 8－5　花窗图案(一)

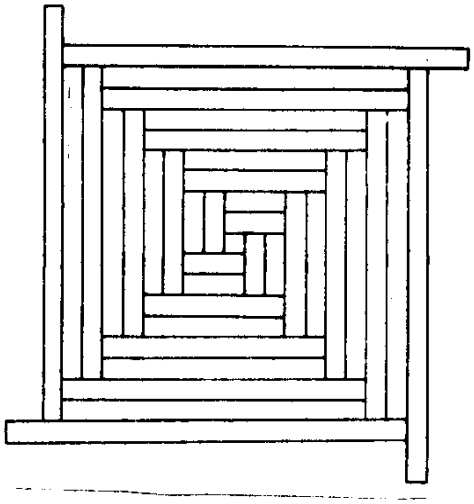


图 8－6　五风车嵌套图

3. 若把程序 No. 8－26 中的 20、30 行改为

```
20 FOR N=1 TO 7 STEP 2
30 FOR L=-N-3 TO N+3
```

可绘出更形象的花窗图案见图 8－7。

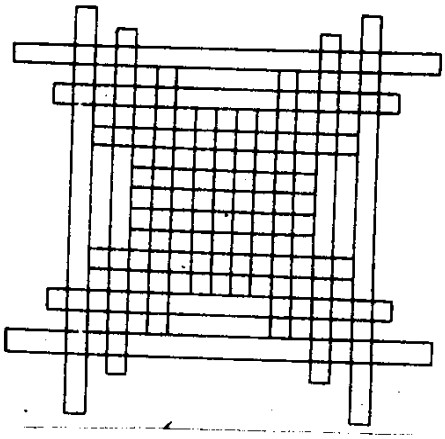


图 8－7　花窗图案(二)

4. 若把程序 No. 8－26 的 20、30 行改为

```
20 FOR N=1 TO 7 STEP 2
30 FOR L=-RND(N+3) TO RND(N+3)
```

则会绘出千变万化的图案，现仅录其三幅，见图 8－8～图 8－10。

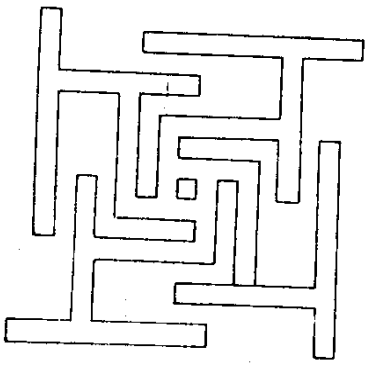


图 8－8

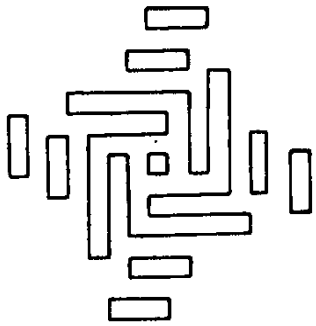


图 8－9

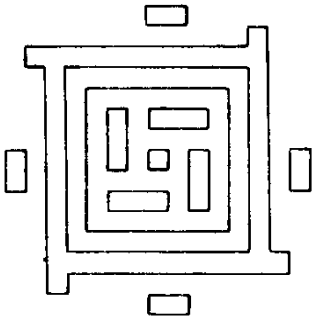


图 8－10

5. 若把程序 No. 8－26 的 20、30 行改为

```
20 FOR N=1 TO 7 STEP RND(2)+1
30 FOR L=－RND(N+2) TO RND(N+2)
```

则可设计出各种墙体的结构图样。见图 8－11，图 8－12。图 8－4 就是使用这一程序绘出的。

由以上改造可见，编程绘图的方法是灵活的。同一个程序，仅仅改变其中的某一个变量，就可使程序的前、后运行结果面目全非。未做以上讨论之前，谁会预见到绘制五套环的程序能够绘出如此千变万化的图案呢？尤其是第五步改造后绘制的图形，将会给建筑设计师们以创造性的启迪。难道它们不是极新颖的墙体结构吗？

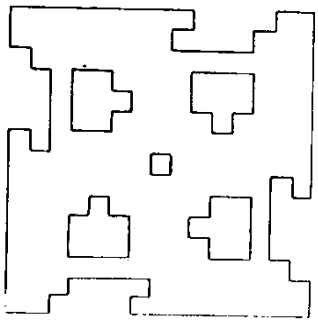


图 8－11

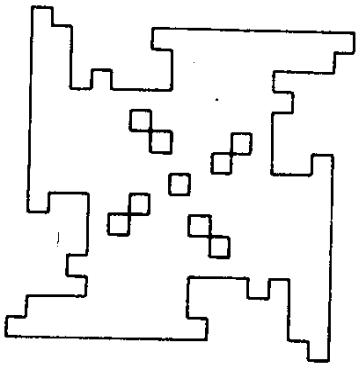


图 8－12

8.3.3 字符函数的妙用

在第五章，我们曾介绍了用字符函数 LEFT $、RIGHT $ 绘制字母三角、字母菱形的程序（见程序 No. 5－7，No. 5－8）。现在介绍 MID $ 函数的绘图应用。

例 8.5 编程打印菱形图案

```
5 REM No. 8－29
10 CLS: X$ ="* * * * * * * * * * * * * *":X=10
20 FOR K=－4 TO 4:Y=Y+1
30 IF K<=0 THEN X=X－1:GOTO 50
```

```
40 X=X+1:LOCATE X,Y
50 PRINT MID$(X$,5-ABS(K),9-2*ABS(K))
60 NEXT
RUN     ↙  打印如图 8-13 的菱形图案
```

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
```

图 8-13 菱形

现在讨论程序 No. 8-29 的变形。

1. 把 50 行改为

```
50 PRINT MID4(X$,5-ABS(K),9-ABS(K))
```

后，运行程序则打印出如图 8-14 的楔形图案。

```
      * * * * *
    * * * * * *
  * * * * * * *
* * * * * * * *
* * * * * * * *
  * * * * * * *
    * * * * * *
      * * * * *
```

图 8-14 楔形

2. 把 50 行改为

```
50 PRINT MID$(X$,5-ABS(K),5-ABS(K))
```

后，运行程序将打印如图 8－15 的三角形。

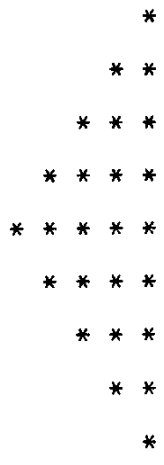


图 8－15　三角形

3. 若对程序 No. 8－29 加 45 行，改 50 行如下：

```
45 IF K=0 THEN PRINT X$ :GOTO 60
50 PRINT MID$ (X$ ,5－ABS(K),1)
```

运行程序则打印如图 8－16 的箭头图案。

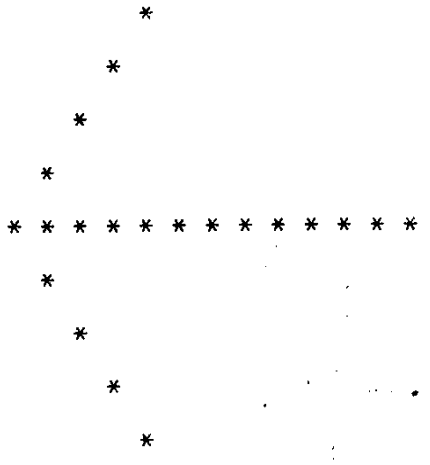


图 8－16　箭头

4. 若对程序 No. 8－29 加 45 行，改 10、50 行

```
10 CLS:X$ ="* * * * *"
45 IF K=0 THEN PRINT X$ ;X$ ;X$ : GOTO 60
50 PRINT MID$ (X$ ,5－ABS(K),9－ABS(K))
```

后，运行程序则打印如图 8－17 的锚形图案。

```
        * * * * *
      * * * *
    * * *
  * *
* * * * * * * * * * * * * * *
  * *
    * * *
      * * * *
        * * * * *
```

图 8－17　锚形

改造程序 No. 8－29 还可以打印多种图案，留给读者练习。

8.3.4　数字图形

用一组有序的数字，构成各种对称的图形，是程序绘图中的又一技巧。

例 8.6　编程打印一多层尖塔，从顶向下依次用顺序自然数组成。

```
5 REM No. 8－30
10 CLS:INPUT "1～7="; N:CLS
20 FOR I=1 TO N
30 LOCATE 15－2 * I,2 * I
40 FOR J=1 TO 2 * I－1
50 PRINT I;
60 NEXT:NEXT
```

RUN　↙　显示

1～7=？□　　键入 5　↙　　打印图 8－18 的图案。

```
        1
      2 2 2
    3 3 3 3 3
  4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5
```

图 8－18　数字宝塔

若运行程序后键入 7 则打印 7 层宝塔。键入 4 则打印 4 层宝

塔。

例 8.7 编程打印数字菱形，要求从外向内的每一层用一类数字围成，外层数最大，中间是“0”。

编程如下：

```
5 REM No. 8－31
10 CLS: INPUT "INPUT 2－6="; N:CLS
20 FOR I=0 TO 2*N
30 K=K+1:LOCATE 7+2*ABS(N－I)－N,K
40 FOR J=N TO ABS(N－I) STEP－1
50 PRINT J;
60 NEXT
70 IF K=1 OR K=2*N+1 THEN 110
80 FOR P=J+2 TO N
90 PRINT P;
100 NEXT
110 NEXT
120 END
```

运行程序，显示

INPUT 2－6=？□ 若键入 3 则打印图 8－19 的图形。

```
        3
      3 2 3
    3 2 1 2 3
  3 2 1 0 1 2 3
    3 2 1 2 3
      3 2 3
        3
```

图 8－19 数字菱形(一)

若运行程序，键入 5 则打印图 8－20 的图形。

```
          5
        5 4 5
      5 4 3 4 5
    5 4 3 2 3 4 5
  5 4 3 2 1 2 3 4 5
5 4 3 2 1 0 1 2 3 4 5
  5 4 3 2 1 2 3 4 5
    5 4 3 2 2 3 4 5
      5 4 3 4 5
        5 4 5
          5
```

图 8－20　数字菱形(二)

由图 8－19、8－20 可见，程序 No. 8－31 达到了例 8－7 的要求。

程序说明：

虽然同是打印菱形，程序 No. 8－31 比程序 No. 8－29 要复杂得多。程序 No. 8－31 采用了两层循环，外循环由 20、30、70、110 四行语句组成，其作用是完成每行的打印定位，其中 70 行的作用是对菱形上、下两个单字符的转向控制。内循环由两个循环程序组成，40～60 行完成菱形左半部的打印，80～100 行完成菱形右半部分的打印。

下面讨论程序 No. 8－31 的变形。

1.　把程序 No. 8－31 中的 40 行改为

```
40 FOR J=N TO 2 * ABS(N－I) STEP－1
```

则打印数字粗箭头，见图 8－21。

```
          5
        5 6
      5 6
    5 4 5
  5 4 3 2 3 4 5
5 4 3 2 1 0 1 2 3 4 5
  5 4 3 2 3 4 5
    5 4 5
      5 6
        5
```

图 8－21　粗箭头

2.　若改 40 行为

```
40 FOR J=N TO 6 * ABS(N－I) STEP －1
```

则打印数字细箭头。见图 8－22。

```
          5
        5 6
      5 6
    5 6
  5 6
5 4 3 2 1 0 1 2 3 4 5
  5 6
    5 6
      5 6
        5 6
          5
```

图 8－22　细箭头

3.　若改 40 行、80 行为

```
40 FOR J=N TO 2 * ABS(N－I) STEP －1
80 FOR P=6 * J+2 TO N
```

运行程序，键入 3，则打印数字小箭头，如图 8－23。

```
        3
      3 14
    3 2 8
  3 2 1 0 -4 -3 -2 -1 0 1 2 3
    3 2 8
      3 14
        3
```

图 8－23　小箭头

还可进行多种变化，留给读者练习。

8.3.5　屏幕万花筒

例 8.8　编程在电视屏幕上演示万花筒。

欲演示万花筒，需在屏幕上的不同位置随机打印不同颜色的不同图形。我们使用随机函数，随机选定打印的图形，随机确定图形的色彩，随机产生图形的打印座标和增量，并使每一次打印的图形在屏幕上的位置上、下、左、右对称，从而使打印的图形既杂乱而又有规律，使画面千变万化、变幻无穷，形成优美、迷人的彩色画卷。实现程序见 No. 8－32。

```
5 REM No. 8-32
10 CLS
20 C=RND(4):C$ =CHR$ (RND(71)+184)
30 X=RND(24)+1:Y=RND(20)+1
40 DX=RND(4):DY=RND(4)
50 IF DX=0 OR DY=0 THEN 20
60 IF X<1 OR X>27 OR Y<1 OR Y>22 THEN 20
70 LOCATE X,Y:PRINT C$ ;:COLOR X,Y,C
80 LOCATE 28-X,Y:PRINT C$ :COLOR 28-X,Y,C
90 LOCATE X,24-Y:PRINT C$ :COLOR X,24-Y,C
100 LOCATE 28-X,24-Y:PRINT C$ :COLOR 28-X,24-Y,C
110 X=X+DX:Y=Y+DY:GOTO 60
120 END
```

RUN ↙

屏幕上对称地显示彩色图形，并很快形成璀璨绚丽的画卷。

8.3.6 图的最短路径

图的最短路径问题已超出绘图程序的讨论范畴。它是计算机在人类作业（生产、运输、邮寄投递等领域）管理中的应用，本应单列一节介绍，但考虑到本文仅对该问题作粗浅的讨论，不深入研究，而且又涉及到图，所以安排在本节最后作一简单介绍，以使读者对计算机应用于人类的生产活动有一个大概的了解。

在生产实践、运输管理、工程建设、邮电投递等社会活动中，诸如工艺线路的安排、货场的布局，管道的铺设，投递路线的选择等等问题，都与寻找一个“图的最短路径”密切相关。图 8－24 给出一个有向图，图中 V1、V2、V3……V7 为图的结点，相邻两点的走向用箭头表示，各箭线旁边所标的数字为相邻两点间的距离。工程上把这种有向图称为网络图，结点表示事件，箭线表示活动。箭线边的数字既可以是距离，也可以是时间，也可以是耗费。任何工程项目的管理都可抽象成图 8－24 的网络图形式。不管箭线边上数字代表何种意义，求各点间的最短路程和路径是网络图的中心目的和任务。

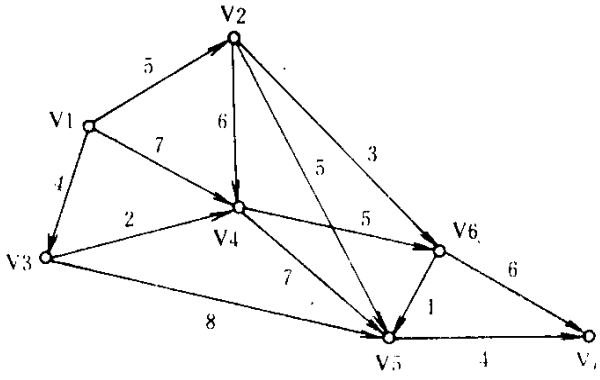


图 8－24　网络图

问题十四：求出 V1 到各结点的最短路程和路径。

1. 算法简介

关于计算网络图中结点间的最短路程和路径问题，狄克斯屈拉(DIKSTRA)于1959年提出了一种计算方法。下面结合图8—24简单介绍。

第一步：首先定义任意两点之间的距离为W(Vi;Vj)(对于图8—24有1≤i≤7,1≤j≤7)。如果结点Vi与结点Vj是相邻的两点并且Vi至Vj的方向与图中所示的方向一致，那么W(Vi,Vj)就等于Vi、Vj箭线边所标的距离如V1～V2有W(V1,V2)＝5，再如V4～V6有W(V4,V6)＝5。反之，若Vi与Vj不是相邻的两点，或者尽管两点相邻但Vi至Vj的方向与图中不一致则规定其距离为无穷大。如W(V2,V1)＝∞，再如V1、V5有W(V1,V5)＝∞。

根据算法规定，对照图8—24，我们可以列出一张7个结点的点间距离表(见表8—2)，表中的第i行第j列就是W(Vi,Vj)的值。

表8—2　网络图各结点间距离表

	j=1	j=2	j=3	j=4	j=5	j=6	j=7	
i=1	∞	5	4	7	∞	∞	∞	
i=2	∞	∞	∞	6	5	3	∞	
i=3	∞	∞	∞	2	8	∞	∞	
i=4	∞	∞	∞	∞	7	5	∞	
i=5	∞	∞	∞	∞	∞	∞	4	
i=6	∞	∞	∞	∞	1	∞	6	
i=7	∞	∞	∞	∞	∞	∞	∞	

第二步：设D(Vi)为所求结点V1至Vi的最短路程。并设A为图中各结点的集合，即有A＝{V1,V2,V3,……V7}。对结点V1给定永久性标号L(V1)＝0，V1到V1的最短路径D(1)＝0；对其它结点则给以试探性标号L(V)＝∞。这时把结点V1置入另一集合$\overline{A}$中，并在集合A中除去V1，V1视作得到永久性标号的结点V^*。

第三步：对集合A中所剩的结点V给以新的试探性标号L

(V)：

$L(V)=\min\{L(V^*)+W(V^*,V),L(V)\}$

第四步：把第三步所得的各 L(V)进行比较，取出其中最小的一个 L(Vi)，把这个结点定为 Vt，使其成为永久性标号的结点 V*，并使其进入集合 $\overline{A}$ 中。并且令 D(Vt)=L(Vt)。

如果集合 A 中的结点都进入 $\overline{A}$ 中，则运算终止。否则，再回到第三步。

这样周而复始，最终可求得 V1 至 Vi(本例 $1\leqslant i\leqslant 7$)的最短路程 D(Vi)。

若要求出 V1 至各点的最短路径，那么在运算过程中，每当集合 A 中的一个结点 V 变为永久性标号时，则记录下产生这个永久性标号的那个已成为永久性标号的结点 Vi。这样要求 V1 至 Vj($1\leqslant j\leqslant 7$)的最短路径，可由 Vj 往前回溯到 V1 就行了。

关于狄克斯屈拉算法的原理已超出本书的范围，有兴趣的读者可参阅有关书籍。读者只要掌握如何填写结点的点间距离表和如何把数据输入程序及如何运用程序求出最短路程和路径就可以了。

根据上面的算法编程见程序 No. 8－33。(注：原程序为王万岳同志所编，运用于 APPLE—Ⅱ型微机。笔者将其移植到电脑游戏学习机中)。

```
5 REM No. 8－33
10 CLS:INPUT "NODES NUM,∞:";X,Y
20 DIM W(X,X)
30 FOR I=1 TO X
40 FOR J=1 TO X
50 READ W(I,J)
60 NEXT:NEXT
70 DATA 99,5,4,7,99,99,99
80 DATA 99,99,99,6,5,3,99
```

```
90 DATA 99,99,99,2,8,99,99
100 DATA 99,99,99,99,7,5,99
110 DATA 99,99,99,99,99,99,4
120 DATA 99,99,99,99,1,99,6
130 DATA 99,99,99,99,99,99,99
1000 DIM V(X),T(X),U(X),M(X),D(X)
1005 DIM E(X),H(X),F(X)
1010 V(1)=0:D(1)=0
1015 PRINT 1"--" 1 "* *D(1)=" D(1)
1020 FOR I=2 TO X
1025 V(I)=Y:M(I)=0:U(I)=0
1030 NEXT
1035 S=1:K=1:N=0:P=1
1040 FOR I=2 TO X
1045 T(I)=N+W(S,I)
1050 IF T(I)<V(I) THEN 1065
1055 U(I)=V(I)
1060 GOTO 1070
1065 U(I)=T(I)
1070 NEXT
1075 N=Y
1080 FOR I=2 TO X
1085 IF N>U(I) THEN N=U(I):S=I
1090 NEXT
1095 D(S)=N
1100 E(1)=1:Z=0:P=P+1:E(P)=S:A=E(P)
1105 FOR I=1 TO P-1
1110 B=E(I):H(A)=D(A)-W(B,A)
1115 IF H(A)=D(B) THEN 1125
1120 NEXT
1125 Z=Z+1:F(Z)=E(I):A=E(I)
```

```
1130 IF A=1 THEN 1140
1135 GOTO 1105
1140 FOR I=Z TO 1 STEP -1
1145 PRINT F(I) "--";
1150 NEXT
1155 PRINT S"**D(" S")="D(S)
1160 M(S)=Y
1165 FOR I=2 TO X
1170 IF,M(I)=Y THEN 1180
1175 V(I)=U(I):GOTO 1185
1180 V(I)=Y
1185 NEXT
1190 K=K+1:IF K>X-1 THEN END
1195 GOTO 1040
RUN    ↙    显示
```

NODES NUM. ,∞:? □提示用户输入结点数和无穷大的数。对于问题十四，有 7 个结点，结点间的最大距离是 8，对无穷大数的选取原则是：取最大距离的 10 倍以上，为便于输入我们选“99”。因此，输入 7 和 99 则打印出如下结果：

```
1--1**D(1)=0
1--3**D(3)=4
1--2**D(2)=5
1--3--4**D(4)=6
1--2--6**D(6)=8
1--2--6--5**D(5)=9
1--2--6--5--7**D(7)=13
```

结果表示第一结点 V1 到 V1 的距离是 0；V1 到 V3 的距离是 4；V1 到 V2 的距离是 5；V1 到 V4 的最短路径是 V1→V3→V4，最短距离是 6；V1 到 V6 的最短路径是 V1→V2→V6，最短距离是 8；V1 到 V5 的最短路径是 V1→V2→V6→V5，最短距离是 9；V1

到 V7 的最短路径是 V1→V2→V6→V5→V7，最短距离是 13。

程序简要说明：

1. 10 行为键盘输入语句。要求依次输入结点的数量和无穷大的数。

2. 30～60 行是把各结点间的距离数据装入数组 W(X,X)中。数据装在 70～130 行语句中。

3. 1015～1070 行的程序完成算法的第三步。

4. 1075～1095 行的程序完成算法的第四步。

5. 1100～1155 行的作用是求 V1 到 Vj(1≤j≤7)的最短路径和路程，并把它们打印出来。

6. 1160～1185 行的程序完成使得到永久性标号的结点 V* 从集合 A 中进入 $\overline{A}$ 中。

7. 1190～1195 行的作用是判断算法是否结束，若没有结束则转 1040 行继续进行第三步。

不要求读者看懂程序，只要掌握该程序的使用就行了。

为加深印象，我们再计算一个实例。

问题十五：网络图如图 8－25，试求出 V1 至各点的最短路径和路程。

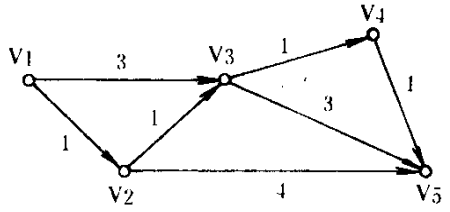


图 8－25　问题十五网络图

根据图 8－25 可列出各点间的距离表。根据表 8－3 把数据填入程序的 DATA 语句中。

```
60 DATA 50,1,3,50,50
70 DATA 50,50,1,50,4
80 DATA 50,50,50,1,3
```

```
90 DATA 50,50,50,50,1
100 DATA 50,50,50,50,50
```

表 8—3　问题十五各点间距离表

	j=1	j=2	j=3	j=4	j=5
i=1		1	3		
i=2			1		4
i=3				1	3
i=4					1
i=5					

注：表中空白处的数为∞。

运行程序，依次输入 5,50 ↙则打印如下结果：

```
1——1 * *D(1)=0
1——2 * *D(2)=1
1——2——3 * *D(3)=2
1——2——3——4 * *D(4)=3
1——2——3——4——5 * *D(5)=4
```

对照图 8—25 验证，运算结果是正确的。

8.4　排序类程序的编写技巧

在人类的社会活动中，经常遇到排序问题。如人们购物、入场、登机、上车时的挨号排队，各种比赛、竞赛、评优、评奖活动中的名次角逐，职工培训、学生升学中的考试成绩统计等等，无不与排序密切相关。排序有很多种类：有的以时间先后为依据；有的以速度快慢为依据，有的以效益高低为依据；有的则以得分多少为依据等等。但是无论何种排序最终都抽象归结为数字的排序。数字排序的一般描述就是：对一定数量的、杂乱无章的数据，按其大小进行排列顺序。可见，排序工作是一项费工费时的、多次重复的、简单而又繁重的脑力劳动。尤其在参与排序的数据量较大时，人工排序的效率极低而出错率较高。因此，电子计算机应用后，人们首先想到

的是使用电脑进行排序。电子计算机的特点之一就是能够快速、准确地进行简单重复的计算，因此，计算机在排序领域中大显身手，得到了广泛的应用。本节着重介绍计算机排序的方法种类和计算机排序在不同领域中应用的编程技巧。

8.4.1 常用的排序方法

计算机排序的方法有几十种，每种方法各有侧重、各有特点、各有利弊、各有优缺点。因此，在不同的排序应用程序的设计中，采用不同的排序方法，会得到不同的排序效果。若选择的方法不当，有时会占用较多内存而排序速度较慢。可见，在不同的程序中，合理使用排序方法，可少占内存，节约时间，提高效率。我们拟选择常用的 14 种排序方法进行介绍，并简要评价各种方法的优、缺点。

为便于比较各方法的排序速度，我们选定统一的排序数据，并规定各种程序的运行时间以按 RUN 键开始计时到打印出全部排序结果为结束。选定数据量为 50 个(即 N＝50)，最小取值 A＝1，最大取值 B＝50。

下面介绍各种排序方法。

1. 转移法

转移法使用了两个一维数组，数组 A(N)用来存放待排序的数，B(N)用来存放排序后的数。转移法排序的编程思路是：首先在 A(N)数组中找到一个最小的数，把它转移到 B(N)数组的最后一个位置，并用一个足够大的数(比 B 大)代替 A(N)中的最小数；然后在 A(N)中找出一个次小的数转移到 B(N)的倒数第二个位置，同样用一个足够大的数代替 A(N)中最小数的位置。如此反复进行，直到处理完 A(N)中的最后一个最小数为止。

程序见 No. 8－34

```
5 REM No. 8－34
10 CLS:N＝50:A＝1:B＝50
20 DIM A(N),B(N)
```

```
30 FOR I=1 TO 50:READ A(I):NEXT
40 S=B+1
50 FOR I=N TO 1 STEP-1
60 A=A(I):B=1
70 FOR J=1 TO N
80 IF A<=A(J) THEN 100
90 A=A(J):B=J
100 NEXT
110 B(I)=A:A(B)=S
120 NEXT
130 FOR I=N TO 1 STEP -1:PRINT B(I) " ";:NEXT
200 DATA 15,30,4,46,27,13,16,20,19,49,50,1,3,9,29,21,2,45,
    38,47,34,18,12,6,14,8,24,35,10,42,5,7,33,31,11,44,36,41,
    37,43,32,17,23,25,36,28,40,48,22,26
```

程序说明：

(1)10～40 行为初始化程序。30 行把 50 个数据读入 A(N)数组。

(2)50～120 行完成排序，转移。

(3)130 行打印输出结果。

运行程序，19.35 秒后打印如下结果

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
```

优缺点评价：

转移法是一种基本的排序方法。

优点：①概念容易理解。

②排序的数字区间没有要求。

缺点：①要建立两个一维数组 A(N)和 B(N)，占用内存多。

②用双循环排序，数据间比较次数多。

③排序速度慢。

2. 枚举法

枚举法只设了一个一维数组 A(N)。枚举法的编程思路是：先拿出数组 A(N)中的第一个数据 A(1)与以后的各数比较，遇到比它小的数则与其交换，交换后继续与后面的数进行比较，再遇到比它小的数，再交换，直到与第 50 个数 A(50)比较完后则 A(1)中装的是最小数，然后再拿 A(2)与以后各数比较。可见枚举法是转移法的改进，与转移法不同的是，比较出的最小数仅在数组本身中进行交换。另外，内循环中循环变量 J 的初值以 I 的取值而递增，所以内循环的次数比转移法减少 50%。程序见 No. 8－35

```
5 REM No. 8－35
10 CLS: N=50:A=1:B=50
20 DIM A(N)
30 FOR I=1 TO N:READ A(I):NEXT
40 FOR I=1 TO N－1
50 FOR J=I+1 TO N
60 IF A(I)>A(J) THEN SWAP A(I),A(J)
70 NEXT:NEXT
130 FOR I=1 TO N: PRINT A(I) " ";:NEXT
200 DATA (同 No. 8－34)
```

运行程序，11. 77 秒后打印与 No. 8－34 相同的结果。

优缺点评价：

优点：①程序结构简单，容易理解。

②排序与数据区间大小无关。

缺点：①使用双重循环排序。

②排序速度仍较慢。

3. 择换法

择换法的编程思路与转移法相似，它先找出 A(N)中的最小数，然后把这个数放在 A(N)的最后一个位置，并把原来的最后一个数放置到最小值原来的位置。然后找出 A(N)中除了最后一个

数以外的最小值，并与A(N)中次后一个数的位置进行交换。如此反复，直到A(N)中的最后一个数处理完为止。见程序No.8－36

```
5 REM No.8－36
10 CLS:N=50:A=1:B=50
20 DIM A(N)
30 FOR I=1 TO N:READ A(I):NEXT
40 FOR I=N TO 2 STEP －1
50 A=A(I):B=1
60 FOR J=2 TO I
70 IF A<=A(J) THEN 90
80 A=A(J):B=J
90 NEXT
100 C=A(I):A(I)=A:A(B)=C:NEXT
130,200(同 No.8－34)
```

运行程序，8.91秒后打印结果。

优缺点评价：

优点：①在同一个数组内排序，节省了内存。

②与数据所在区间大小没有关系。

③排序速度比1、2两种方法快。

缺点：使用双重循环，数据比较次数仍较多。

4.气泡法：

气泡法与枚举法的编程思路相同，不同的是使大数排在前面，小数排在后面，程序见No.8－37。

```
5 REM No.8－37
10、20、30(同 NO,8－36)
40 FOR I=N TO 2 STEP －1
50 FOR J=1 TO I－1
60 IF A(J)>=A(J+1) THEN 80
70 SWAP A(J),A(J+1)
80 NEXT
```

```
90 NEXT
```

130、200(同 No. 8－36)

运行程序，10. 88 秒后打印结果

优缺点评价：

优点：①在同一数组中排序，占用内存少。

②与区间大小无关。

缺点：①要用双重循环排序。

②排序速度慢。

5. 跳跃法：

跳跃法的编程思路是：引入跳跃步长变量 B 控制跳跃跨度，使步幅两端的变量相互比较。比较中，把较小的数交换到前面的变量中。开始步长 B=N/2，即把参加排序的数分为两部分，A(0)～A(N/2)和 A(N/2+1)～A(N－1)然后拿 A(0)和 A(N/2)比，之后再拿 A(1)和 A(N/2+1)比，直到 A(N/2)和 A(N－1)比较完毕后，把步长缩小一倍，即 B=B/2，再进行第二轮比较交换。比较中，小数逐渐前移，大数逐渐后退。最后当 B=1，比较结束后，则排序完成程序见 No. 8－38。

```
5 REM No. 8－38
10 CLS: N=50:A=1:B=50
20 DIM A(N)
30 FOR I=0 TO N－1:READ A(I):NEXT
40 B=N/2
50 IF B<=0 THEN 140
60 C=B
70 IF C>=N THEN 130
80 D=C－B
90 IF D<0 THEN 110
100 IF A(D)>A(D+B) THEN 120
110 C=C+1:GOTO 70
```

```
120 SWAP A(D),A(D+B):D=D-B:GOTO 90
130 B=B/2:GOTO 50
140 FOR I=0 TO N-1:PRINT A(I) "  ";:NEXT
200 DATA(同上)
```

运行程序，9.22 秒后打印结果。

优缺点评价：

优点：①使用一维数组，少占内存。

②与区间大小无关。

缺点：①虽然没有使用循环语句，但 GOTO 语句使用得太多，程序不好理解。

②比较次数仍然较多，排序速度改善不大。

6. 速升法：

速升法的编程思路与跳跃法相似，不同的是引入了双重循环，速度要慢一些程序见 No. 8－39。

```
5 REM No. 8-39
10 CLS: N=50:A=1:B=50
20 DIM A(N)
30 FOR I=1 TO N:READ A(I):NEXT
40 K=N
50 K=(K+1)/2
60 FOR I=1 TO K
70 S=0
80 FOR J=K+I TO N STEP K
90 IF J>N THEN 130
100 IF A(J-K)>=A(J) THEN 120
110 SWAP A(J),A(J-K):S=S+1
120 NEXT
130 IF S<>0 THEN 70
140 NEXT
150 IF K<>1 THEN 50
```

```
160 FOR I=N TO 1 SETP -1:PRINT A(I) " ";:NEXT
200 DATA(同上)
```

运行程序，13.9 秒后打印结果。

优缺点评价：

优点是同一数组中排序，占用内存少；与数据区间大小没有关系。

缺点是采用了双重循环排序，虽然用相距为 K 的两项对比可减少比较次数，但比单循环的比较次数要多，故排序速度较慢。

7. 交替法

交替法是对气泡法的改进。它在进行奇数次处理时，从前往后对数组 A(N)中的相邻两数进行比较处理，使较大的数排在前面；在进行偶数次处理时，从后往前进行相邻两数的比较，使较大的数排在前面程序见 No. 8－40。

```
5 REM No. 8-40
10～30(同上)
40 FOR I=1 TO N-1
50 K=(I-1)/2+1:IF I MOD 2=0 THEN 110
60 FOR J=K TO N-K
70 IF A(J)>=A(J+1) THEN 90
80 SWAP A(J),A(J+1)
90 NEXT
100 GOTO 150
110 FOR J=N-K TO K+1 STEP -1
120 IF A(J)<=A(J-1) THEN 140
130 SWAP A(J),A(J-1)
140 NEXT
150 NEXT
160、200(同 No. 8-39)
```

运行程序，16 秒后打印结果。

优缺点评价：

优点：同一维数组排序，少占用内存，排序与数据所在的区间无关。

缺点：需要用双层循环，而且在第一层循环中套了两个单循环，所以排序速度很慢。

8. 两极法

两极法的编程思路是：第一轮比较 A(1)与 A(N)，若 A(N)＞A(1)，则 A(1)再与 A(N－1)比较，若仍有 A(N－1)＞A(1)，再与 A(N－2)比较，若 A(N－2)＜A(1)则交换数值，把较小的数放入 A(1)。然后固定 A(N－2)，拿 A(2)与其比较，若 A(N－2)＞A(2)，再拿 A(3)与其比较，若 A(N－2)＜A(3)则交换内容，把较小的数放入 A(3)。再固定 A(3)不变拿 A(N－3)与 A(3)比较，若 A(N－3)＞A(1)，再拿 A(N－4)与其比较。如此，比较的两数值逐渐从两极靠近中间。第一轮比较结束后，较小的数值排在数组的前半部分，较大的数值在后边。然后再对 A(1)→A(2)、A(2)→A(3)……之间进行比较调整。调整后进行第二轮比较交换，直至排序结束。程序见 No. 8－41。

```
5 REM No. 8－41
10 CLS: N=50:A=1:B=50
20 DIM A(N),Z(N)
30 FOR I=1 TO 50:READ A(I):NEXT
40 I=1:Z(I)=N+1:Q=1:H=N
50 GOSUB 110:I=I+1:Z(I)=X
60 IF X－Q>1 THEN H=X－1:GOTO 50
70 Q=Z(I)+1:I=I－1:IF I<0 THEN 100
80 IF Z(I)－Q>1 THEN H=Z(I)－1:GOTO 50
90 GOTO 70
100 FOR I=1 TO N:PRINT A(I) " ";:NEXT:END
110 L=Q:R=H
```

```
120 IF R-L=0 THEN X=R:RETURN
130 IF A(R)<A(L) THEN SWAP A(R),A(L):L=L+1:GOTO 150
140 R=R-1:GOTO 120
150 IF R-L=0 THEN X=R:RETURN
160 IF A(R)<A(L) THEN SWAP A(R),A(L):R=R-1:GOTO 120
170 L=L+1:GOTO 150
180 END
```

运行程序，10.87 秒后打印结果。

优缺点评价：

优点：两极法属跳跃法的改进型，使用变量 I 控制循环次数，从两极向中间逐渐缩小比较范围，循环比较次数比双重循环少、排序速度稍快。排序不受数据所在区间限制。

缺点：①使用两个数组，占用内存较多。

②程序不易阅读。

9. 比照法

比照法的编程思路是：建立一个数组 B(M)，它的大小和需排序数区间相对应(M=B-A)。然后把数组 A(N)中的数取出一个，按其大小比照到 B(M)数组中去，实际上就是通过比照，记录下各个数值出现的次数程序见 No. 8-42。

```
5 REM No. 8-42
10 CLS: N=50:A=1:B=50
20 DIM A(N),B(B-A)
30 FOR D=1 TO N:READ A(D):NEXT
40 FOR I=1 TO N:B(A(I)-A)=B(A(I)-A)+1:NEXT
50 FOR I=B TO A STEP -1
60 IF B(I-A)=0 THEN 90
70 PRINT I" ";:B(I-A)=B(I-A)-1:GOTO 60
80 PRINT:PRINT
90 NEXT
```

200 DATA(同上)

程序中 40 句排序，50～90 句为输出程序。

运行程序，仅 4 秒钟便打印完结果。

优缺点评价：

优点：①用单循环排序，排序速度最快。

②特别适用于要排序的数比较多，数据所在区间(A，B)不特别大的情况。

③程序结构简单。

缺点：①要建立两个数组，占用内存相对要多一些。

②受数据区间(A，B)的影响大。当(A，B)大而 N 较小时，效果不好。

10. 映射法

映射法的编程思路：首先把数组 A(N)中的数，逐个映射到存放相同数据的数组 D(M)中去(其中 M＝B－A，B 为数据区间最大值，A 为最小值)，映射出相同数据的个数。然后利用辅助数组 K(M)排出名次。最后将有名次的数据再一次映射到辅助数组 T(N)中去，输出 T(N)就得到排序结果。程序见 No. 8－43。

```
5 REM No. 8－43
10 CLS: N＝50:A＝1:B＝50:H＝B－A＋1
20 DIM A(N),T(N),D(H),K(H):M＝0
30 FOR I＝1 TO N: READ A(I):D(A(I))＝D(A(I))＋1:NEXT
40 FOR I＝H TO 1 STEP －1:IF D(I)＝0 THEN 60
50 K(I)＝M＋1:M＝M＋D(I)
60 NEXT:K(0)＝M＋1
70 FOR I＝1 TO N:A＝A(I):T(K(A)＋D(A)－1)＝A:D(A)＝D(A)
   －1:NEXT
80 FOR I＝N TO 1 STEP －1:IF T(I)＝0 THEN 100
90 PRINT T(I)"  ";
100 NEXT
```

```
200 DATA(同上)
```

程序中 40～70 行为排序程序，80～100 行为输出程序。

运行程序，7.4 秒后打印完结果。

优缺点评价：

优点：只用单循环排序，排序速度快。

缺点：①建立了四个数组，占用内存较大。

②受数据所在区间(A，B)影响较大，当(A，B)大时不宜使用。

11. 下标法：

下标法是比照法的变形。它的编程思路是：把要排序的每个数作为变量的下标，该数就存放在该下标变量中。如 33 存在 A(33)中，26 存在 A(26)中等，然后把不为零的下标变量中的数据打印出来，排序过程即告结束程序见 No. 8－44。

```
5 REM No. 8－44
10 CLS: N=50 :
20 DIM A(N)
30 FOR I=1 TO 50:READ S:A(S)=S:NEXT
40 FOR I=1 TO N
50 IF A(I)<>0 THEN PRINT A(I) "  ";
60 NEXT
200 DATA(同上)
```

程序中 40～60 行为排序输出程序。

运行程序，2.6 秒后打印完结果。

优缺点评价：

优点：①使用一个数组，占用内存少。

②排序与输出共用一个循环程序，故运行速度最快。

缺点：①排序受数据大小限制，数据较大时占用内存较多。

②对重复的数据只能排出一个，其它则丢掉。

12. 合并法

合并法的编程思路是：定义一个两维数组 A(1,N)。把要排序的数据装入 A(0,N)数组中，然后由 A(0,N)数组提供数据，每次拿出相邻的两个数据进行比较，如 A(0,X)、A(0,X+1)。

若A(0,X)>A(0,X+1)　则令

A(1,X)=A(0,X)、A(1,X+1)=A(0,X+1)；

若A(0,X)<A(0,X+1)　则令

A(1,X)=A(0,X+1)、A(1,X+1)=A(0,X)

第一轮比较结束后，A(1,N)数组中的数据排列是“高、低、高、低、高、低……”的顺序。

第二轮则由 A(1,N)提供数据，每次拿出相邻的四个数据进行比较，以“最大、大、小、最小”的顺序依次装入 A(0,N)数组的对应单元中。第二轮比较结束后，A(0,N)数组中的数据顺序是“高、次高、较低、低、高、次高、较低、低、高……”

同理，下一轮比较，又由 A(0,N)提供数据，每次拿出八个相邻数据进行比较，以高在前、低在后的顺序依次存入 A(1,N)的对应单元中。

这样，每比较一轮，排好顺序的数据量增大一倍。可以推出，合并法进行比较的轮次(Y)为

Y=log2N

可见，合并法的排序速度是较快的。

程序见 No. 8－45

```
5 REM N0. 8－45
10 CLS: N1=50:N=1:U=1
20 N=2*N:IF N<=N1 THEN X=X+1:GOTO 20
30 N=N/2:DIM A(1,N)
40 FOR I=1 TO N:READ A(V,I):NEXT
50 T=1:FOR I=1 TO X:T=2*T
60 FOR J=1 TO N－T+1 STEP T
70 A=J:B=A+T/2:C=B－1:D=A+T－1
```

```
80 GOSUB 110:NEXT
90 SWAP U,V:NEXT
100 FOR I=N TO 1 STEP -1:PRINT A(V,I)"  ";:NEXT:END
110 Z=A-1
120 Z=Z+1:IF A(V,A)<A(V,B) THEN 150
130 A(U,Z)=A(V,A):A=A+1:IF A<=C THEN 120
140 M=D:L=B:GOTO 170
150 A(U,Z)=A(V,B):B=B+1:IF B<=D THEN 120
160 M=C:L=A
170 FOR W=L TO M:Z=Z+1
180 A(U,Z)=A(V,W):NEXT:RETURN
```

程序说明：

(1)10～40 为初始化程序。其中由 20 行求出比较轮次数 X。

(2)100 行为输出语句。

(3)50～90 行为轮次控制程序。

(4)110～180 行为比较、交换程序。

运行程序，6.1 秒排出 32 个数据。

优缺点评价：

优点：①由于使用 2 的指数控制比较轮次，故比较次数少，速度较快。

②排序不受数值区间限制。

缺点：①由于使用 2 的指数控制轮次，所以排序的数据个数必须为 2 的幂数。如为 8、16、32、64、128 个等。

②程序量较大。

③使用两维数组，占用内存稍多。

13. 无交换法

无交换法的编程思路是：定义四个相同的数组 A(N)、B(N)、R(N)、L(N)。用数组 A(N)装欲排序的数据。当两个数进行比较时，如 A(3)与 A(5)比较，若 A(3)>A(5)，则把这一比较结果记

入 R(N)数组，令 R(3)＝5；若 A(3)＜A(5)则记入 L(N)数组，令 L(3)＝5。当全部比较完毕后，L(N)、R(N)的有关变量中记录了比较的结果，然后用 B(N)数组的有关变量控制调用 R(X)或 L(X)，令 R(X)或 L(X)作 A(N)数组的变量的下标，从而按数值的大小选择打印，使输出结果成为排好顺序的数程序见 No. 8－46。

```
5 REM No. 8－46
10 CLS: N＝50:DIM A(N),B(N),R(N),L(N)
20 FOR I＝1 TO N:READ A: NEXT
30 FOR I＝2 TO N:Z＝1
40 IF A(I)＞A(Z) THEN 70
50 IF L(Z)＝0 THEN L(Z)＝I:GOTO 90
60 Z＝L(Z):GOTO 40
70 IF R(Z)＝0 THEN R(Z)＝I:GOTO 90
80 Z＝R(Z):GOTO 40
90 NEXT:Z＝1:X＝1
100 IF L(Z)＝0 THEN 120
110 B(X)＝Z:X＝X＋1:Z＝L(Z):GOTO 100
120 PRINT A(Z)"  ";:IF R(Z)＜＞0 THEN Z＝R(Z):GOTO 100
130 X＝X－1:IF X＝0 THEN END
140 Z＝B(X):GOTO 120
```

程序中 10～20 行为初始化程序；30～90 行完成数据的比较及对 L(N)、R(N)的有关变量赋值；100～140 行为输出打印程序。

运行程序，14.22 秒后打印完结果。

优缺点评价：

优点：排序不受数据区间限制。

缺点：使用数组多，占用内存量大，排序速度慢。

14. 插入法

插入法排序的思路是：把 A(N)数组中的数一个一个地取出，插入到相应的正确位置上去，使它前面的数都不比它小，而它后面

的数都不比它大，程序见 No. 8－47。

```
5 REM No. 8－47
10 CLS: INPUT "N,A,B="; N,A,B:DIM A(N)
20 FOR I=1 TO N:READ A(I):NEXT
30 FOR I=2 TO N:A=I－1:J=A
40 IF A(I)>=A(J) THEN 60
50 K=J:B=A(I):GOTO 80
60 J=J－1:IF J>0 THEN 140
70 K=0:B=A(I)
80 IF A=K THEN 120
90 FOR J=A TO K+1 STEP －1
100 A(J+1)=A(J):NEXT
110 A(K+1)=B
120 NEXT
130 PRINT:FOR I=1 TO N:PRINT A(I) " ";:NEXT
140 DATA(同上)
```

程序中 30～120 行为排序程序。

运行程序，15. 44 秒钟打印完排序结果。

优缺点评价：

优点：①在同一数组中排序，占用内存少。

②与数据区间的大小无关。

缺点：由于插入时要把后面的各项向后移动一位，因而增加了程序运行时间，排序速度较慢。

以上介绍了 14 种排序方法，现将各方法的特点、性能汇总于表 8－4。

从表 8－4 可见，这些排序方法各有特点，有的占用内存少，有的排序速度快，有的对数据区间有要求。因此，在应用时要根据排序数据情况、内存情况选用合理的排序方法，以求得较好的排序效果。

表 8－4　十四种排序方法比较

排序方法	速度 N＝50	使用数组	循环层数	与数据区间关系
转移法	19.35	A(N),B(N)	双 1＋1	无
交替法	16	A(N)	双 1＋2	无
插入法	15.44	A(N)	双 1＋GOTO	无
无交换法	14.22	A(N),B(N),R(N),L(N)	双 1＋GOTO	无
速升法	13.9	A(N)	双 1＋1	无
枚举法	11.77	A(N)	双 1＋1	无
气泡法	10.88	A(N)	双 1＋1	无
两极法	10.87	A(N),Z(N)	双 GOTO＋GOTO	无
跳跃法	9.22	A(N)	双 GOTO＋GOTO	无
择换法	8.91	A(N)	双 1＋1	无
映射法	7.4	A(N),T(N),D(M),K(M)	两单	有
合并法	6.1	A(1,N)	双 1＋1	无
比照法	4	A(N),A(M)	单	有
下标法	2.6	A(N)	单	有

8.4.2　排序应用(一)

统计学生成绩

1. 单科成绩排序

学生成绩的统计工作是每所学校在期末考试后都要投入大量的人力来做的一件事情，这种工作要求仔细，有耐心，工作量很大。运行 No.8－48，可把你从繁琐枯燥的数字排列束缚中解放出来。

```
5 REM No.8－48
10 INPUT "N＝"; M:DIM T(M),N(M),A(M):CLS
20 FOR I＝1 TO M:READ T(I),N(I),A(I):NEXT
30 FOR I＝1 TO M－1
40 FOR J＝I＋1 TO M
50 IF T(I)＞＝T(J) THEN 70
60 SWAP T(I),T(J):SWAP N(I),N(J):SWAP A(I),A(J)
70 NEXT:NEXT
```

```
80 FOR I=1 TO M-1:FOR J=I+1 TO M
90 IF T(I)<>T(J) OR N(I)<N(J) THEN 110
100 SWAP A(I),A(J):SWAP N(I),N(J)
110 NEXT:NEXT
120 FOR I=1 TO M-1:FOR J=I+1 TO M
130 IF T(I)<>T(J) OR A(I)<=A(J) THEN 150
140 SWAP A(I),A(J):SWAP N(I),N(J)
150 NEXT:NEXT
160 FOR I=1 TO M:PRINT I;
170 LOCATE 5,CSRLIN:PRINT T(I);:LOCATE 10,CSRLIN:
    PRINT N(I)"  "A(I):NEXT
180 DATA (得分)90,(学号)1010,(年龄)19
    (为简化程序,以后一行写多组数据)
190 DATA 90,1011,20,88,1047,24,97,1063,20,87,1053,18,88,
    1060,17,47,1070,18,90,1078,20,90,1048,22,87,1038,18
200 DATA 88,1001,19,97,1002,19,100,1003,18,90,1100,23,90,
    1700,21,90,1111,20,90,1701,22
```

程序说明：

(1)30～70 行为成绩排序程序。成绩按降序排列。

(2)80～110 行为学号排序程序。学号按升序排列。

(3)120～150 行为年令排序程序。年龄按升序排列。

(4)160～170 行为输出程序。

(5)排序原则为：相同成绩中，年龄小的在前；相同成绩、相同年龄中，学号小的在前。

运行程序，键入“17”，则打印结果如下：

```
 1        100        1003        18
 2        97         1002        19
 3        97         1063        20
 4        90         1010        19
 5        90         1011        20
```

6	90	1078	20
7	90	1111	20
8	90	1700	21
9	90	1048	22
10	90	1701	22
11	90	1100	23
12	88	1060	17
13	88	1001	19
14	88	1047	24
15	87	1038	18
16	87	1053	18
17	47	1070	18

程序 No. 8－48 要开辟三个数组，占用内存较多，在 V3 版本下最多可排 1000 名学生的成绩在 BS. 2A 下最多可排 600 人，而且排序速度太慢。另外，从排序结果来看，相同分数的人，名次不同，显然也不太合理。程序 No. 8－49 较好地弥补了上述的缺陷。

```
5 REM No. 8－49
10 INPUT "N,M="; N,M:DIM A$(M),A(M):CLS (N 为考生人
   数,M 为满分)
20 FOR I=1 TO N:READ X$,X:A(X)=A(X)+1:A$(X)=A
   $(X)+X$:NEXT
30 FOR I=M TO 0 STEP －1:IF A(I)=0 THEN 70
40 IF A(I)>=2 THEN 90
50 FOR J=1 TO A(I):C=C+1:PRINT MID$(A$(I),4*(J－1)+
   1,4),C,I
60 NEXT
70 NEXT
80 END
90 FOR J=1 TO A(I):C=C+1:PRINT MID$(A$(I),4*(J－1)+
   1,4),C,I
```

```
100 C=C-1:NEXT
110 C=C+1:GOTO 70
120 DATA 8801,96,8802,98,8803,97,8804,98,8805,120
130 DATA 8806,98,8807,99,8808,98,8809,99,8810,95
```

运行程序，键入人数 10 人，键入满分数 120，打印如下结果。

```
8805        1        120
8807        2        99
8809        2        99
8802        3        98
8804        3        98
8806        3        98
8808        3        98
8803        4        97
8801        5        96
8810        6        95
```

程序说明：

(1)10 行为人机对话和初始化程序。其中 N 为考生人数，M 为满数。N，M 可视实际情况灵活选用，但应和 DATA 语句中存储的记录一致。

(2)20 行完成各数据的读入。其中 X 记录不同考生的分数，兼作数组 A(X)、A$(X)的下标。其中 A(X)=A(X)+1，统计考分为 X 的学生人数，是数值相加。A$(X)=A$(X)+X$，是存储相同成绩考生的学号，是字符串连接。如本例中考 99 分的有两个人，学号分别为 8807、8809，经过 20 行处理后，A(X)=A(99)=2，则 A$(99)=88078809。

(3)30～110 行程序根据不同的条件，分别判断名次，打印排序的学号、名次和成绩。其中 50～60 行打印 A(I)=1 的排序结果。90～100 行打印 A(I)>1 的排序结果(并列名次处理)。变量 C 记录名次。

本程序占用内存少，执行速度快，不受考生数量的限制，无论是30人还是300人，占用内存量不增加。另外，修改循环变量的初、终值，还可以进一步提高速度。如单科成绩最高分为100、最低为60分，则M取100，同时30句改为FOR I=100 TO 60 STEP −1……。这样循环次数大大减少，执行速度可进一步提高。本程序较好地处理了相同分数的名次问题。

程序No. 8−49的唯一缺点是，受字符串变量最多为31个字符的限制，所以考生成绩相同的人数最多不能超过7人。若把学号改为3位数，则可增加到10人。

2. 多科成绩排序

有时要对学生的多门成绩进行排序，还要求出总分和平均分，程序No. 8−48，No. 8−49显然不能胜任。程序No. 8−50可以较好地完成这一工作。

```
5 REM No. 8−50
10 INPUT "Ren shu=";N (N=人数)
20 M=3:CLS     (M=科目数)
30 DIM A(N,M),B(N),C(N),C(N),D(N),V(N)
40 PRINT "   CHENG JI TONG JI BIAO"
50 R$="m c x m x h yw sx wl Z F PJ"
60 Q$="−−−:−−−:−−−:−−:−−:−−:−−−:−−"
70 FOR I=1 TO N:READ F$,C(I)
80 FOR J=1 TO M:READ A(I,J):B(I)=B(I)+A(I,J):NEXT
90 D(I)=B(I):V(I)=D(I)/M:NEXT
100 PRINT Q$:PRINT R$:PRINT Q$
110 RESTORE
120 FOR I=1 TO N:FOR J=1 TO N
130 IF B(I)<B(J) THEN 150
140 SWAP B(I),B(J)
150 NEXT:NEXT
```

```
160 FOR I=1 TO N:IF B(I)=B(I-1) THEN 230
170 FOR K=1 TO N:READ F$,C(K),P,P,P
180 IF B(I)<>D(K) THEN 220
190 PRINT I"  "F$; C(K);
200 FOR J=1 TO M:PRINT A(K,J);: NEXT
210 PRINT D(K); V(K);:PRINT:PRINT
220 NEXT:RESTORE
230 NEXT
240 END
250 DATA D J,801,78,90,87  (DJ=杜  杰)
260 DATA LZX,802,56,67,78  (李自香)
270 DATA L Y,803,98,87,76  (梁  燕)
280 DATA DXG,804,98,97,96  (达希古)
290 DATA W Y,805,76,75,74  (吴  洋)
300 DATA WHW,806,64,35,67  (翁号文)
310 DATA QZQ,807,67,50,67  (裘兆启)
320 DATA P J,808,64,63,62  (平  姜)
```

注:各考生的姓名仅取汉语拼音的字头。

程序说明:

(1)数组和变量:N 为学生人数;M 为考试科目数;I 为按总分排列的名次;F$ 为学生的姓名变量;A(N,M)数组装 N 个学生的 M 门成绩;B(N)和 D(N)为学生的总成绩;C(N)为学号;V(N)为平均成绩。

DATA 语句中,第一个是学生姓名,第二是学生的学号,后面是语文、数学、物理各科的成绩。

(2)10～30 行为人机对话和初始化程序。

(3)40～60 行为成绩表的表头程序。

(4)70～150 行语句为读入数据和排序程序。

(5)160～230 行为输出程序。

运行程序，输入“8”，则打印如下成绩统计表（见表 8－5）。

3. 综合成绩排序

(1)一般排序程序

在诸如智力竞赛等活动，要对参赛者进行口试和笔试两项考试，因此要对两项成绩进行综合排序。例如笔试成绩占总分 60%，口试成绩占 40%，试编程计算总分并排出名次。程序见 No. 8－51。

表 8－5　学生成绩统计表

CHENG JI TONG JI BIAO

mc	xm	xh	yw	sx	wl	zf	pj
1	DXG	804	98	97	96	291	97
2	LY	803	98	87	76	261	87
3	DJ	801	78	90	87	255	85
4	W Y	805	76	75	74	225	75
5	P J	808	78	60	70	208	69
6	LZX	802	56	67	78	201	67
7	QZQ	807	67	50	67	184	61
8	WHW	806	64	35	67	166	55

注：mc—名次，　xm—姓名，　xh—学号，　yw—语文，
　　sx—数学，　wl—物理，ZF—总分，　PJ—平均分数。

```
5 REM No. 8－51
10 INPUT "N=";N:DIM A(N),B$(N),D(N),E(N),F(N),G(N),S
   (N):CLS
20 FOR I=1 TO N:A(I)=I
30 READ B$(I),D(I),E(I),F(I),G(I)
40 S(I)=(6*F(I)+4*G(I)+5)/10:NEXT
50 FOR I=1 TO N－1:FOR J=1+I TO N
60 IF S(I)>=S(J) THEN 80
70 SWAP A(I),A(J):SWAP B$(I),B$(J):SWAP D(I),D(J):
   SWAP E(I),E(J):SWAP F(I),F(J):SWAP G(I),G(J):SWAP S
   (I),S(J)
```

```
80 NEXT:NEXT
90 PRINT "k h x m nl b j bs ks zf mc":PRINT
100 FOR I=1 TO N:S$=STR$(A(I)):S$=RIGHT$(S$,LEN
    (S$)-1)
110 PRINT RIGHT$("00"+S$,3)" "B$(I);D(I);E(I);F(I);G
    (I);S(I)" "I
120 NEXT:END
500 DATA Z L,15,102,92,77
510 DATA XJW,16,201,84,79
520 DATA TXY,14,101,94,86
530 DATA CDY,16,303,95,90
540 DATA L T,17,301,80,85
```

运行程序，键入 N＝5，则打印如下结果：

kh	xm	nl	bj	bs	ks	zf	mc
（考号	姓名	年龄	班级	笔试	口试	总分	名次）
004	CDY	16	303	95	90	93	1
003	TXZ	14	101	94	86	91	2
001	Z L	15	102	92	77	86	3
002	XJW	16	201	84	79	82	4
005	L T	17	301	80	85	82	5

程序说明：

①10 行人机对话，定义数组。其中：数组 A(N)记录考生的考号；B$(N)记录考生的姓名；D(N)记录考生的年龄；E(N)记录考生的班级；F(N)记录笔试成绩；G(N)记录口试成绩；S(N)记录总分。

②20～40 行读入数据，计算每个考生的总分。

③50～80 行完成排序。

④90～120 行输出结果。

程序 No. 8－51 的排序效率是较低的。原因是所有的数据项都要随着总分的交换而交换。另外，它还缺乏处理并列名次的能

力，例如同为82分的两名学生名次却不相同。

(2)高效率的排序方法——考号索引法

我们知道减少数据交换量是提高排序效率的关键。若采用以考号为索引项的方法进行排序，可以避免考号以外的任何数据交换，从而可大大简化处理过程，提高排序效率。程序见No. 8－52。

```
5 REM No. 8－52
10 INPUT "N=";N:DIM A(N),B$(N),D(N),E(N),F(N),G(N),S
   (N),H(N):CLS
20 FOR I=1 TO N:A(I)=I
30 READ B$(I),D(I),E(I),F(I),G(I)
40 S(I)=(6*F(I)+4*G(I)+5)/10:NEXT
50 FOR I=1 TO N－1:L=I:FOR J=I+1 TO N
60 IF S(A(L))<S(A(J)) THEN L=J
70 NEXT:SWAP A(I),A(L)
80 GOSUB 180:NEXT:GOSUB 180
90 PRINT "  TABLE 1 ":PRINT
100 GOSUB 200
110 FOR I=1 TO N: M=A(I):P=H(M):GOSUB 160:NEXT:
    PRINT
120 PRINT "  TABLE 2 ":PRINT
130 GOSUB 200
140 FOR I=1 TO N:M=I: P=H(I):GOSUB 160:NEXT:PRINT
150 END
160 S$=STR$(M):S$=RIGHT(S$,LEN(S$)－1)
170 PRINT RIGHT$("00"+S$,3)" "B$(M);D(M);E(M);F(M);
   G(M);S(M)" "P:RETURN
180 K=K－(S(A(I))<>S(A(I－1)))
190 H(A(I))=K:RETURN
200 PRINT "k h x m nl b j bs ks zf mc":PRINT:RETURN
500～540 DATA (同 No. 8－51)
```

运行程序，输入“5”，则打印两张成绩统计表，其中“TABLE 1”按名次打印，“TABLE 2”按原顺序打印。

TABLE 1 （按名次排序）

kh	xm	nl	bj	bs	ks	zf	mc
004	CDY	16	303	95	90	93	1
003	TXZ	14	101	94	86	91	2
001	Z L	15	102	92	77	86	3
002	XJW	16	201	84	79	82	4
005	L T	17	301	80	85	82	4

TABLE 2 （按原顺序打印）

kh	xm	nl	bj	bs	ks	zf	mc
001	Z L	15	102	92	77	86	3
002	XJW	16	201	84	79	82	4
003	TXY	14	101	94	86	91	2
004	CDY	16	303	95	90	93	1
005	L T	17	301	80	85	82	4

程序说明：

本程序以各考生号为自然数的特点，在比较总分时，仅交换各存储考号的下标变量 A(L)与 A(J)的值(60、70 行)，其它数组一律不动，这样仅在数组 A 中记载排序结果，其它数组变量中的内容不动，因此比 No. 8－51 的数据交换量减少了近 90%。同时，该程序还较好地解决了并列名次问题。另外，还能以两种不同方式输出排序结果，以满足不同的需要。

程序 20～40 行把数据装入相应数组，并计算总分装入 S 数组；50～80 为排序程序；90～110 行输出“TABLE 1”；120～140 行输出“TABLE 2”；160～170 行为打印各输出项子程序；180～190 行为名次变量 P 的计算子程序；200 行为表头各栏条目打印子程序。

(3)高效率的排序方法——映射索引法

考号索引排序能大大减少排序时的数据交换次数，提高了排序效率。但是，这一方法仍然离不开多数排序法的重要手段——反复比较。因此，当记录条数较多时仍然要花费较多的时间。利用映射索引排序则可以同时避免数据交换和数据比较这两个过程，因而具有很高的排序效率，程序见 No. 8－53。

```
5 REM No. 8－53
10 INPUT "N="; N:DIM M(100):CLS
20 FOR I=1 TO N:READ A$,C,D,X1,X2
30 S=(6*X1+4*X2+5)/10:M(S)=M(S)+1:NEXT
40 FOR I=100 TO 0 STEP －1
50 K=K+SGN(M(I)):M(I)=K:NEXT:RESTORE
60 PRINT "k h x m nl b j bs ks zf mc":PRINT
70 FOR I=1 TO N:READ A$,C,D,X1,X2:S=(6*X1+4*X2+
   5)/10
80 S$=STR$(I):S$=RIGHT(S$,LEN(S$)－1)
90 PRINT RIGHT("00"+S$,3)" "A$;C;D;X1;X2;S;M(S)
100 NEXT:END
500～540 DATA (同上)
```

运行程序，键入"5"，打印如下结果：

kh	xm	nl	bj	bs	ks	zf	mc
001	Z L	15	102	92	77	86	3
002	XJW	16	201	84	79	82	4
003	TXY	14	101	94	86	91	2
004	CDY	16	303	95	90	93	1
005	L T	17	301	80	85	82	4

程序 No. 8－53 仅用一个单循环就完成了排序过程，无需任何数据交换和比较，所以其排序效率高于其他任何排序方法。该程序的主要特点是数组 M(100)的使用。程序的 30 行中，以总分 S 为索引项，用数组 M(S)存放各种分数的个数；而在 50 行中又用 M(I)存放各个分数的名次，同时也解决了并列名次的问题。这种

利用同一数组先后存放两类相关数据的技巧，在一般程序中很少见，也可以说，这是本程序的一个创造，这也充分说明了灵活地使用下标去管理数组的意义。这种排序方法只须根据总分的大小，定义一个固定容量的数组 M(100)。这样，可节省大量的存储空间。排序不受记录数量的限制，可以一次处理几十万个数据记录（当然要以内存能容下 DATA 数据为前提）。同时，本程序使用简单变量取代数组还可进一步提高排序速度。因此，程序 No. 8－53 是一个很值得推广的学生成绩排序程序。

8.4.3 排序应用(二)

电视大奖赛比分统计

目前，国内外举行的各种歌手、体育、演唱比赛的评分办法，都是由若干个评分委员评分，然后去掉一个最高分，去掉一个最低分，求出其它几个评分的算术平均数。该平均数即为选手的得分。这种评分办法很合理，也很普及。使用我们的游戏电脑也可以为各种比赛进行电脑计分，排列名次，其速度和效果可与高档微机相媲美。下面给出两个程序供读者选用。

1. 电脑评分程序(一)

程序清单见 No. 8－54。

```
5 REM No. 8－54
10 INPUT "ping wei shu=" ; N:DIM A(N)
20 FOR I=1 TO N:PRINT "Di" I " hao ping wei ";: INPUT G
30 A(I)=G:NEXT:S=0
40 P=0
50 FOR J=1 TO N－1:IF A(J)>=A(J+1) THEN 70
60 SWAP A(J),A(J+1):P=1
70 NEXT
80 IF p<>0 THEN 40
90 FOR K=2 TO N－1:S=S+A(K):NEXT
100 PRINT "less a max=";A(1):PRINT "less a min="A(N):PRINT
```

```
110 PRINT "De fen=" S:PRINT
120 PRINT "Xia yi ge xuan shou ":GOTO 30
```

运行程序，打印

ping wei shu=? □　　（评委数）

如有7个评委，则键入"7"，随后打印

Di 1 hao ping wei ? □

假如各评委的评分依次为99、98、86、78、89、92、88，依次输入以上各数后则打印

less a max=99　　（去掉一个最高分=99）

less a min=78　　（去掉一个最低分=78）

De fen=89　　（最后得分=89）

Xia yi ge xuan shou　　（下一个选手）

程序No.8－54可方便地计算出各选手的得分，但不能排序，这是该程序的不足。

注意，若评分以10分制为满分时，也要把评分乘10以后再输入电脑。

2.电脑评分程序（二）

程序清单见No.8－55

```
5 REM No.8-55
10 CLS: INPUT "xuan shou shu="; M:DIM F(M),H(M)
20 INPUT "ping wei shu="; N
30 FOR A=1 TO M:MAX=0:MIN=100:H(A)=A
40 PRINT "* * * * * * * * * * * * * * * * * * * * * * * * * *":
   PRINT A" hao xuan shou de fen":PRINT
50 FOR B=1 TO N: PRINT B" hao ping wei";:INPUT P
60 IF P>MAX THEN MAX=P
70 IF P<MIN THEN MIN=P
80 F(A)=F(A)+P:NEXT
90 F(A)=(F(A)-MAX-MIN)/(N-2):PRINT
100 PRINT "less a max " MAX:PRINT "less a min" MIN:PRINT A"
```

```
    hao zui hou de fen" F(A):PRINT "==========
    ==========="
110 NEXT
120 FOR I=1 TO M-1:FOR J=I+1 TO M
130 IF F(J)>F(I) THEN SWAP F(I),F(J):SWAP H(I),H(J)
140 NEXT:NEXT
150 PRINT "==========================
    =":PRINT
160 FOR I=1 TO M:PRINT "Di"I"ming" H(I) " hao" F(I) " fen"
170 NEXT:PRINT
180 PRINT "=========================="
190 END
```

运行程序，显示

xuan shou shu= ?□ 假如选手有五名，则键入“5” ↙ 显示

Ping wei shu= ?□ 假如有七名评委，则键入“7” ↙ 则打印

```
* * * * * * * * * * * * * * * * * * * * * * *
1 hao xuan shou de fen
1 hao ping wei ?□
```

假如1号评委评份97，则键入97 ↙ 又询问2号评委的评分，假如以后各评委的评分依次是87、76、95、98、77、88，依次输入后，将显示图8－26的画面。

```
* * * * * * * * * * * * * * * * * * * * * * *
1 hao xuan shou de fen
1 hao ping wei ? 97
2 hao ping wei ? 87
3 hao ping wei ? 76
4 hao ping wei ? 95
5 hao ping wei ? 98
6 hao ping wei ? 77
```

7 hao ping wei ? 88

less a max 98

less a min 76

1 hao zui hou de fen 88

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

＊ ＊

2 hao xuan shou de fen

1 hao ping wei ? □

图 8－26

图中的汉语拼音分别为：

“1 号选手得分

1 号评委 97 分

……

去掉一个最高分 98

去掉一个最低分 76

1 号最后得分 88”

随后又做好了 2 号选手评分的准备。若七个评委对以后四名选手的评分依次为：

2 号 88、97、86、78、98、65、77

3 号 96、89、72、77、86、91、80

4 号 65、76、88、79、92、89、87

5 号 75、99、98、97、99、96、95

待对以后四名选手评分完毕后，立即打印名次：

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

di 1 ming 5 hao 97 fen

di 2 ming 1 hao 88 fen

di 3 ming 2 hao 85 fen

di 4 ming 4 hao 83 fen

di 5 ming 3 hao 82 fen

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

程序说明：

①10 行人机对话，要求输入选手数 M，定义数组 F(M)装入各选手得分，H(M)数组记录选手号数。

②20 行人机对话，要求输入评委数 N。

③30～110 行完成各选手的评分输入和最后得分的计算、打印。

④120～140 行完成名次排序。

⑤150～180 行输出比赛结果。

程序 No. 8－55 是一个较全面的评分程序，但愿它能给你的工作或娱乐带去方便。

8.4.4 排序应用(三)

电话号码册

1. 简单的电话号码册

每人都有许多朋友，都有一本电话号码册，当电话号码较多时，查找某个朋友的电话号码极为不便，若把朋友的姓名按第一个汉语拼音顺序排序，在查找时将会十会方便。程序 No. 8－56 给你提供了这一工具。

```
5 REM No. 8－56
10 CLS: INPUT "INPUT TOTAL NUMBER=";K
20 CLS: DIM A$(K),A(K)
30 FOR I=0 TO K:READ A$(I)
40 B$=LEFT$(A$(I),1):A(I)=ASC(B$)
50 NEXT
60 FOR L=0 TO K－1
70 FOR N=0 TO K－L－1
80 IF A(N)<=A(N+1) THEN 100
90 SWAP A(N),A(N+1):SWAP A$(N),A$(N+1)
100 NEXT:NEXT
110 PRINT "NAME                    PHONE NUMBER:":PRINT
```

```
120 FOR I=0 TO K: PRINT A$(I):NEXT
130 DATA ZHANG XIAO MING  327654,LI  HUA  781274.
140 DATA WANG  AI  HUA  872357,ZHAO  ZHENG HUA
    871326,QIAN  SHAN  QIANG 626351
```

注：为使输出整齐，人名按 18 个字符定位，写入 DATA 语句，见 130、140 行示例。

运行程序打印如下结果：

```
NAME                    PHONE  NUMBER
LI              HUA     781274
QIAN    SHAN    QIANG   626351
WANG    AI      HUA     872357
ZHANG   XIAO    MING    327654
ZHAO    ZHENG   HUA     871326
```

程序说明：

①10～20 行为人机对话，初始化程序，其中 10 行要求输入 DATA 语句中的全部电话号码数。

②30～50 行为读入数据语句，30 行把各个电话号码装入 A$(N)中，40 行把每个人名第一个字母的 ASCII 码装入 A(N)中。

③60～100 行为排序程序。

④110～120 行为输出程序。

注—使用程序 No. 8－56 时，必须把每一个电话号码写入 DATA 语句中，并在 10 行输入所有电话号码的数量。

2. 具有检索功能的电话号码册

具有检索功能的电话号码册程序见 No. 8－57。

```
5 REM No. 8－57
10 CLS: INPUT "How many:";Z:DIM A$(Z,2),B(Z):PRINT
20 FOR I=1 TO Z:READ A$(I,1),A$(I,2):NEXT
30 FOR E=1 TO Z:S$=LEFT$(A$(E,1),1):B(E)=ASC(S$):
   NEXT
```

```
40 FOR F=1 TO Z-1
50 FOR G=1 TO Z-1
60 IF B(G)<B(G+1) THEN 80
70 SWAP B(G),B(G+1):SWAP A$(G,1),A$(G+1,1):SWAP A
   $(G,2),A$(G+1,2)
80 NEXT:NEXT
90 INPUT "please choose:LIST(1),LOOK(2),END(3). INPUT "1,2,
   3 ";X
100 IF X=3 THEN END
110 IF X=1 THEN CLS:GOTO 170
120 CLS:INPUT "NAME";T$:PRINT
130 FOR R=1 TO Z
140 IF T$=A$(R,1) THEN PRINT T$,A$(R,2):PRINT:GO-
    TO 90
150 NEXT
160 PRINT "No this name!":PRINT:GOTO 90
170 PRINT " NAMES ";: LOCATE 20, CSRLIN: PRINT
    "NUMBERS":PRINT
180 FOR Q=1 TO Z:PRINT A$(Q,1);:LOCATE 20,CSRLIN:
    PRINT A$(Q,2):PRINT:NEXT:PRINT:PRINT
190 GOTO 90
200 DATA Zhang xiao ming,327654,Li hua,781274,Wang ai hua,
    872357,Zhao zheng gua,871326,Qian shan qiang,626351
```

运行程序，电脑询问：

How many:? □ （多少电话号码）

键入数字键“5”，回车，则打印

Please choose:LIST(1),LOCK(2),END(3). Input 1,2,3. ? □

（请选择：列出电话号码表(1)，查看某一个电话号码(2)，退出(3)。输入 1,2,3）

若键入数字键“1”，回车，则打印电话号码表如下：

NAMES	NUMBERS
Li hua	781274
Qian shan qiang	626351
Wang ai hua	872357
Zhang xiao ming	327654
Zhao zheng hua	871326

若键入数字键“2”，回车，则清屏，询问

NAME? □　　（输入查找的姓名）

如键入“Wang ai hua”，则打印

Wang ai hua 872357

若键入数字键“3”，回车，则停机。

程序说明：

(1)10 行人机对话，要求输入所有电话号码的数量。语句中定义数组 A$(Z,2)，其中 A$(Z,1)记录各个姓名，A$(Z,2)记录各个电话号码；定义数组 B(Z)记录各姓名第一个字母的 ASCII 码，以便排序。

(2)20 行把数据读入 A$ 数组中。

(3)30 行完成把各姓名的第一个字母的 ASCII 码装入 B(Z)数组中。

(4)40～80 行完成排序。

(5)90 行列出选择菜单供用户选择。若列电话号码表则键数字键“1”；若查询某人的电话号码则键入数字键“2”；若结束则键“3”键。

(6)120～160 行是查询的处理程序。

(7)170～180 行是列表程序。

程序 No. 8－57 的使用说明：

(1)每次使用时必须输入所有电话号码的数量。

(2)所有电话号码都要写入 DATA 语句中。

(3)数据格式：人名为一个数组，号码是一个数据，成对输入。

8.5 辅助教学类程序的编写技巧

游戏电脑除具有以上章节介绍的功能外，还可以辅导中、小学生进行各门功课的练习。本节仅介绍加、减、乘、除、英语单词练习五个简单程序，以作示范。

8.5.1 加法练习程序

1. 两位数加法

程序见 No. 8－58

```
5 REM No. 8－58
10 G=0:CLS:PRINT " BGGIN! ":PRINT
20 FOR I=1 TO 10
30 A=RND(90)+10:B=RND(90)+10
40 PRINT A"+"B"=";
50 INPUT C: IF C<>A+B THEN 70
60 LOCATE 14,CSRLIN－1:PRINT "GOOD!":G=G+10:GOTO 80
70 LOCATE 20,CSRLIN－1:PRINT "WRONG!"
80 NEXT
90 PRINT:PRINT "Your score is" G
100 END
```

程序 No. 8－58 兼有出题，批改、评分三重功能。每次运算 10 题，每计算一题批改一次。若算对则打印"GOOD!"；否则打印"WRONG!"。计算 10 题后，打印得分。

说明：

①30～40 行出题。

②50 行检查计算结果，若算错转 70 行打印"WRONG!"。

③60 行打印正确的评语"GOOD!"。

④90 行打印得分。

⑤在 60、70 行中均使用了 CSRLIN 语句，其目的是对评语的

输出位置定位，使每题的评语打印在该题之后。

2. 单位数多位数加法

通过阅读程序 No. 8－58 可知，只要修改 30 行 RND 函数的基数，就可以任意调整加法的位数以适应不同的年龄阶层。如：

```
30 A=RND(9)+1:B=RND(9)+1
```

则为单位数加法，可适用一年级及学龄前儿童。

```
30 A=RND(900)+100:B=RND(900)+100
```

可进行 3 位数的加法练习。

3. 连加练习

当进行连加练习时，可在 30 行再加一句对 C 赋值的程序。如 C=RND(90)+10。所有 A+B 语句的后面再添“+C”就可以了。

8.5.2 减法练习程序

把 No. 8－58 稍加改造就得到减法练习程序。改造中要注意，30 行中 B 的取值要小于 A，(由于小学生尚无负数的概念)。程序见 No. 8－59。

```
5 REM 8－59
10 G=0:CLS:PRINT "  BEGIN!":PRINT
20 FOR I=1 TO 10
30 A=RND(90)+10:B=RND(A－10)+10
40 PRINT A"－"B"=";
50 INPUT C: IF C<>A－B THEN 70
60 LOCATE 14,CSRLIN－1:PRINT "GOOD!":G=G+10:GOTO 80
70 LOCATE 20,CSRLIN－1:PRINT "WRONG"
80 NEXT
90 PRINT:PRINT "Your Score is";G
100 END
```

本程序与 No. 8－58 结构相同，读者自行分析。

8.5.3 乘法练习程序

程序清单见 No. 8－60

```
5 REM No. 8－60
10 CLS:PRINT "Hello! Let's learn multiplication"
20 S=0
30 FOR I=1 TO 10
40 A=RND(90)+10:B=RND(9)+1
50 PRINT " No. :";I
60 PRINT A" "CHR$(181);B"=";
70 INPUT C:IF C<>A*B THEN LOCATE 20,CSRLIN－1:PRINT
   "X":GOTO 100
80 LOCATE 20,CSRLIN－1:PRINT "V"
90 S=S+10
100 NEXT
110 PRINT "you score is :" S
120 IF S>=90 THEN PRINT "Very good!"
130 PRINT "I hope you will succeed!"
140 PRINT "Do you want to continue? (Y/N)"
150 A$=INKEY$(0):IF A$<>"N" THEN 20
160 PRINT "See you later!"
170 END
```

键入程序，即可进行两位数乘一位数的乘法练习。若欲改变难度可调整 40 行的数值。

8.5.4 除法练习程序

由于 F BASIC 不能进行精确的除法运算，(即没有小数)，所以，对于随机产生的被除数和除数，必须保证能够整除，才能保证商的精确。程序见 No. 8－61。

```
5 REM No. 8－61
10 CLS:PRINT "Hello! Let's learn division"
20 S=0
30 FOR I=1 TO 10
40 A=RND(90)+10:B=RND(8)+2:C=A*B
```

```
50 PRINT "No. :";I
60 PRINT C" " CHR$(182);B"=";
70 INPUT A: IF A<>C/B THEN LOCATE 20,CSRLIN-1:PRINT
   "X":GOTO 100
80 LOCATE 20,CSRLIN-1:PRINT "V"
90 S=S+10
100 NEXT
110 PRINT"You score is:"S
120 IF X>90 THEN PRINT "Very good!"
130 PRINT "Do you want to continue? (Y/N)"
140 A$=INKEY$(0):IF A$<>"N" THEN 20
150 CLS:LOCATE 8,10:PRINT "See you later"
160 END
```

8.5.5 英语单词拼写练习程序

单词拼写是学习英语的一个重要环节，程序 No. 8－62 可以辅导学生进行单词拼写练习。运行程序，屏幕上出现一个不完全的单词，然后要求学生输入缺少的字母，将这个单词拼写完整。如果输入的内容正确，则显示"GOOD!"，接着再出下一个单词的练习。如果输入的答案不正确，将显示"ERROR. REPEAT"，要求你重新输入。如果还是不对，电脑将告诉你正确的答案。程序所出题目的数量由 DATA 语句中的单词决定。做完一次练习后，还可以根据自己掌握单词的情况选择重做或结束。程序清单如下：

```
5 REM No. 8－62
10 CLS:INPUT "N="; N:DIM A$(N),A(N),B$(20)
20 FOR I=1 TO N:READ A$(I):NEXT
30 FOR K=1 TO N
40 X=RND(N)+1
50 FOR J=1 TO K-1
60 IF X=A(J) THEN 40
70 NEXT
```

```
80 A(K)=X:A=LEN(A$(X)):FOR I=1 TO A:B$(I)=MID$(A
   $(X),I,1):NEXT
90 A1=RND(A-1):IF A1<2 THEN 90
100 FOR I=1 TO A1:PRINT B$(I);: NEXT
110 T$=" ":FOR I=A1+1 TO A:T$=T$+B$(I):NEXT:
    PRINT
120 T=0
130 INPUT "Your answer is" ;F$
140 IF F$=T$ THEN PRINT "GOOD!":GOTO 170
150 T=T+1:IF T<2 THEN PRINT "ERROR. REPEAT!":GOTO
    130
160 PRINT "The answer is"; A$(X)
170 NEXT
180 PRINT "Again? (Y/N)": INPUT T$:IF T$="Y" THEN 30
190 END
200 DATA THERE, HERE, DATA, YEAR, GOTO, ANSWER,
    SCHOOL,RESISTOR,CIRCATE,HAVE
```

RUN ↙ 显示

N= ?□ 键入"10" ↙显示

THE

Your answer is?□ 若键入"RE"↙

GOOD!

RESIS

Your answer is?□ 若键入"TOR"↙

GOOD!

GO

Your answer is?□ 若键入"G"↙

ERROR. REPEAT!

Your answer is?□ 若键入"O"↙

The answer is GOTO

SCHO

Your answer is? □

……

电脑就象一个耐心的老师，不厌其烦的出题要你回答，回答不对，会告诉你答案，再出题。你可把你不太熟练的单词输入DATA语句，反复练习，定会加强记忆，掌握迅速。但愿电脑当好你的学习助手。

8.6 键盘练习程序的编写技巧

学习电脑的第一步就是学会键盘操作。而键盘操作的中心内容就是掌握英文打字。由于几乎所有的计算机高级语言程序都是用英文或其简写形式编制的，而电脑键盘各键的排列又都是按英文打字机的键位规则设计的，所以，只有掌握了英文打字才能迅速准确地向电脑输入程序，完成各种运算。

键盘操作既是一项基本技能，又是一门艺术。只有掌握了要领、方法，才能达到迅速准确的目的。我们学习电脑，首先就要养成标准的击键习惯(通常称为指法)。关于英文打字的学习有专业书籍介绍，本节仅简介指法规则和给出几个键盘练习程序，以辅导初学者熟悉键盘、掌握击键规则，提高程序的输入速度，从小养成良好的习惯，为今后使用电脑打下良好的基础。

8.6.1 基本指法介绍

同弹钢琴一样，操作电脑也要讲究指法。初接触电脑的朋友都有这样的感觉：键盘上各键长短不一、排列不齐，且各字母键不按A、B、C、D……的顺序排列，记忆困难、查找不便，使用起来很别扭。实际上，这种排键方式是最科学的选择，它把使用最频繁的字母作为基本键，安排在击键最方便的位置，这是前人从多种排键方案中优选出的最佳方案。我们要迫使自己适应、习惯这种排键方式。

那么，人有十个指头，键盘有几十个按键，它们是怎样分工的呢？一般在英文打字教科书中各指的分工如图 8－27，8－28 所示。

图 8－27：在键盘上画出七道斜线，把键盘分成八个区，每区由一个指头负责。

图 8－28：示意出每个指头分工的字符或按键。

指法规定：A、S、D、F、J、K、L；八键为基本键。要求击键前，八个手指要准确地放在基本键的上方。无论敲击何键，每击一键后都应返回基本键位。另外从开始起就要养成不看键盘击键的好习惯。如欲输入上面那个字符时，若这个键在左手区时，应先按右边的 Shift 键不放，再击字符键，反之亦然。

最后说明键盘输入时的姿式：要求坐姿端正，头稍前倾，大臂自然下垂，上体与键盘相距 20 厘米，手指自然弯曲，轻放于基本键上，两拇指轻放于空格键上。两膝平放，两脚着地，座位高低适中，使两肘与键盘平齐，双眼视线落在键盘右侧或左侧的原稿上，有节奏地敲击各键。

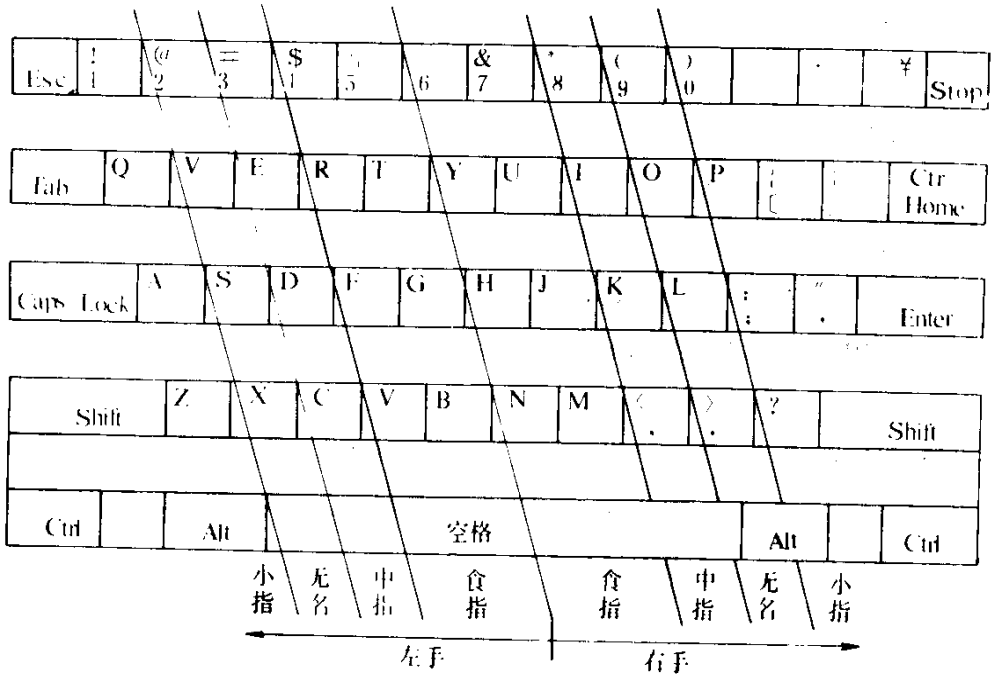


图 8－27　键盘分区图

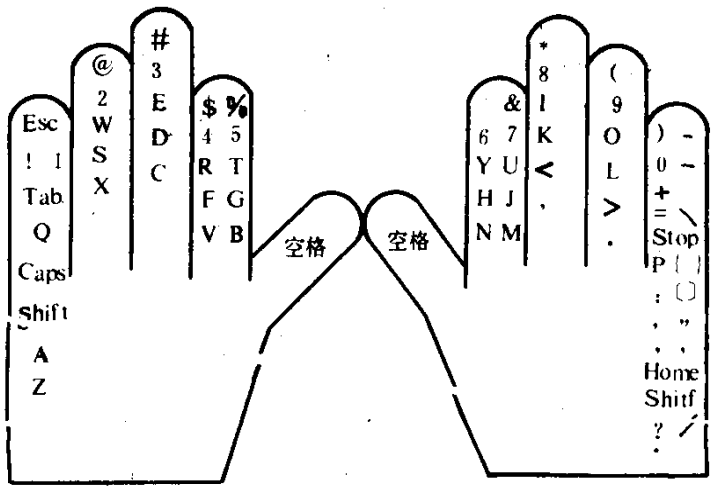


图 8－28

初学者应多用一些时间练指法，做到熟练准确，提高工作效率。

8.6.2 熟悉键盘程序

程序 No. 8－63 是专为初学者编制的熟悉键盘程序。它可以帮助初学者尽快熟悉键盘上各字符键的位置。程序清单如下：

```
5 REM No. 8－63
10 CGEN 3:CGSET 1,1:SPRITE ON
20 VIEW:PLAY "T104C1"
30 PO$ ="0102030405":PP$ ="CDEFGAB"
40 DIM P(58):FOR I=0 TO 58:READ P(I):NEXT
50 DEF SPRITE 0,(0,0,0,0,0)=CHR$ (199)
60 VIEW:K=0:LOCATE 2,5:FOR I=9 TO 0 STEP －1:PRINT I;:
   NEXT
70 PAUSE 100:A$ =CHR$ (RND(43)＋48):SPRITE 0
80 T=T＋1:LOCATE 21,16:PRINT T:LOCATE 25,10:PRINT " ":
   LOCATE 11,10:PRINT A$
90 GOSUB 280
100 FOR I=0 TO 75
110 FOR J=0 TO 5:B$ =INKEY$ :LOCATE 25,10:PRINT B$
```

```
120 SPRITE 0,38+2*I,53
130 IF B$<>" " THEN SWAP A$,B$:GOSUB 210:SWAP A$,B
    $:IF A$=B$ THEN I=200:J=I
140 NEXT:NEXT
150 IF I>100 THEN PLAY "04B1AG2FE3D4C":K=K+1:GOTO 70
160 LOCATE 2,15:PRINT "Right is" K
170 LOCATE 3,16:PRINT "Try again?";: A$=INKEY$(0):IF A
    $="N" THEN LOCATE 0,18:END
180 GOTO 60
190 DATA 0,21,22,23,24,24,25,25,26,27,14,14,5,28,6,7,29,21,
    22,23,24,24,25,25,26,27
200 DATA 14,14,5,28,6,7,21,7,3,2,9,16,10,10,11,19,11,12,13,
    4,4,20,21,14,17,8,17,18,3,15,1,18,0
210 IF A$<" " THEN PLAY "OICIEC":A$=" "
220 IF A$=" " THEN RETURN
230 PO=P(ASC(A$)-32)/7
240 PP=P(ASC(A$-32)-PO*7
250 PLAY MID$(PO$,PO*2+1,2)+MID$(PP$,PP+1,1)
260 PLAY MID$(PP$,PP+1,1)
270 RETURN
280 FOR J=0 TO 5:GOSUB 230:NEXT:RETURN
```

键入程序后，按图 8－29 键控绘图。绘图后，运行程序，显示如图 8－30 的画面。由图 8－30 可见，程序把画面分为五个区，各区的意义如下：

Ⅰ区：时间提示区。运行程序后，在该区降序打印 9、8、7……0，10 个数字用以提示时间的剩余量。在数字的上面有一面小旗从左向右移动，操作者必须在小旗移动到 0 之前键入Ⅱ区显示的字符。否则，将转结束处理。

Ⅱ区：字符提示区。要求操作者键入与该区提示相同的字符。

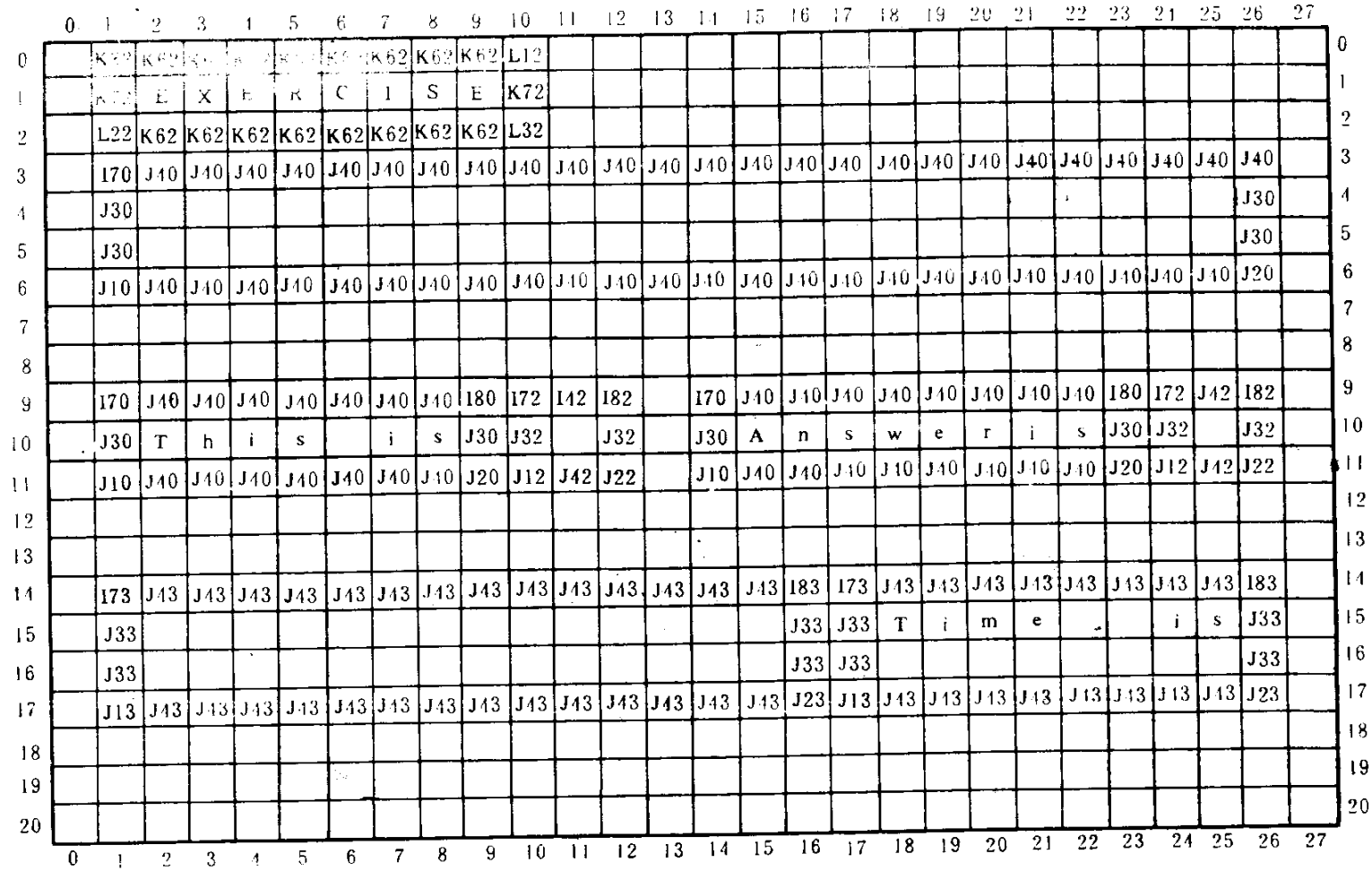

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
0 K62 K62 K62 K62 K62 K62 K62 K62 K62 L12
1 K72 E X E R C I S E K72
2 L22 K62 K62 K62 K62 K62 K62 K62 K62 L32
3 I70 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40
4 J30 J30
5 J30 J30
6 J10 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J20
7
8
9 I70 J40 J40 J40 J40 J40 J40 J40 I80 I72 I42 I82 I70 J40 J40 J40 J40 J40 J40 J40 J40 I80 I72 J42 I82
10 J30 T h i s i s J30 J32 J32 J30 A n s w e r i s J30 J32 J32
11 J10 J40 J40 J40 J40 J40 J40 J40 J20 J12 J42 J22 J10 J40 J40 J40 J10 J40 J40 J40 J40 J20 J12 J42 J22
12
13
14 I73 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 I83 I73 J43 J43 J43 J43 J43 J43 J43 J43 I83
15 J33 J33 J33 T i m e i s J33
16 J33 J33 J33
17 J13 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J43 J23 J13 J43 J43 J43 J43 J43 J43 J43 J43 J23
18
19
20

図 8—29

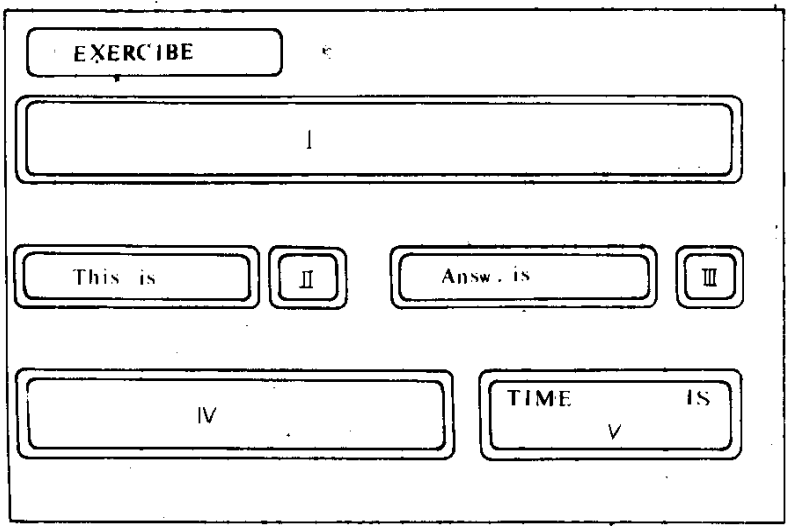


图 8－30

Ⅲ区：答案区。如果操作者键入了正确的字符，则在该区显示该字符。

Ⅳ区：结束选项区。若超过时间（小旗移动到 0）未键入正确的字符，则在该区显示以上的成绩，并在下行显示“Try again?”询问是否继续练习，若结束按“N”键；否则，按任一键后继续练习。

Ⅴ区：练习次数提示区。提示键盘练习进行到第几个字符。

程序说明：

①10～50 行为初始化程序。其中 30、40 两行定义变量 PO $、PP $ 和数组 P(58)用于与 190～280 行一起演奏音乐。

②60 行用于在 I 区打印 9～0，10 个数字。

③70 行随机产生要求输入的字符。

④80 行累计练习的次数并把它们打印在屏幕上。

⑤100～140 行使小旗从左向右移动，并在 110 行读入操作者输入的字符，检查是否与电脑产生的字符相同，若相同则结束循环，转键对处理程序。

⑥150 行为键对处理程序。键对后，累计键对次数，返回 70 行进行下一次练习。

⑦160～180 行为结束处理程序。160 行打印键对的次数，170 行询问是否继续练习，若结束按“N”键。按其它键则返回 60 行继续练习。

⑧190～280 行为音乐演奏程序。选定不同的字符，演奏不同的音乐。各字符和音乐的关系见表 8－6。

表 8－6

字符	音乐	字符	音乐	字符	音乐
0	05D	?	04C	N	03G
1	04C	@	02C	O	03B
2	04D	A	01C	P	04C
3	04E	B	01F	Q	03C
4	04F	C	02E	R	03F
5	04F	D	03E	S	02D
6	04G	E	02E	T	03F
7	04G	F	02F	U	03G
8	04A	G	02F	V	01F
9	04B	H	03G	W	03D
:	03C	I	02A	X	01D
;	03C	J	02G	Y	03G
<	01A	K	02A	Z	01C
=	05D	L	01B		
>	01A	M	01G		

⑨欲提高小旗的运行速度，可令 110、280 行 J 的循环终值为 0。若仍嫌速度不够快，可调整 100、120 行 I 的终值和 I 的倍数。

8.6.3 指法练习程序

键盘熟悉后就可进行指法练习了。指法练习一般是逐指练习，待各指击键准确熟练后，再综合练习。从击键中，我们已经体会到，两个小手指最不听使唤。因此，要对小手指加强练习，直到各指均灵活自如。我们把小手指排为第一，无名指为第二、中指第三、食指第四，编写了指法练习程序 No. 8－64。程序运行后，首先由操作者选择练习的指头(1、2、3、4)，指头选定后再选择速度(即每个字符

显示的时间),一般先选择低速,熟练后再提高速度。然后,根据闪烁字符击中相应键。练习中要始终注意,只看屏幕,不看键盘,根据屏幕上的字符位置由手指记忆住各个键位,以养成不看键盘的习惯。一般在程序 No. 8－64 的辅导下,上机 8～16 个小时就可达到要求。

程序清单如下:

```
5 REM No. 8－64
10 VIEW:SPRITE ON:CGEN 3
20 DIM B(40),A(1,40):S=100
30 FOR I=1 TO 40:READ B(I):NEXT
40 FOR I=0 TO 1:FOR J=1 TO 40:READ A(I,J):NEXT:NEXT
50 DEF SPRITE 0,(0,0,0,0,0)=CHR$(255)
60 GOSUB 160
70 N=－(Z=1)*(RND(8)+1)－(Z=2)*(RND(8)+9)－(Z=3)*
   (RND(8)+17)－(Z=4)*(RND(16)+25)
80 B$=CHR$(B(N))
90 FOR I=S TO 1 STEP －1:LOCATE 24,10:PRINT I" ":BEEP
100 IF F=1 THEN SPRITE 0:F=0:PAUSE 10
110 SPRITE 0,A(0,N),A(1,N):F=1
120 A$=INKEY$:A=ASC(A$)
130 IF A=27 THEN GOSUB 160
140 IF A$=B$ THEN K=K+1:LOCATE 5,10:PRINT K" ":I=0
150 NEXT:GOTO 70
160 LOCATE 23,4:C$=INKEY$(0):Z=VAL(C$):IF ASC(C$)
    =26 THEN 260
170 IF Z<1 OR Z>4 THEN 160
180 LOCATE 25,6:C$=INKEY$(0):C=ASC(C$)
190 IF C<26 OR C>29 THEN 180
200 IF C=28 THEN S=S－10
210 IF C=29 THEN S=S+10
```

```
220 LOCATE 15,8:PRINT S"  "
230 IF C=27 THEN RETURN
240 IF C=26 THEN 260
250 GOTO 180
260 SPRITE 0:CLS:PRINT"GOOD BYE!!!":LOCATE 0,22:END
270 DATA 49,81,65,90,48,80,59,47,50,87,83,88,57,79,76,46,51,
    69,68,67,56,73,75,44,52,82,70,86,53,84,71,66,54,89,72,78,
    55,85,74,77
280 DATA 32,40,48,56,176,184,192,200,48,56,64,72,160,168,
    176,184,64,72,80,88,144,152,160,168,80,88,96,104,96,104,
    112,120,112,120,128,136,128,136,144,152
290 DATA 118,134,150,166,118,134,150,166,118,134,150,166,
    118,134,150,166,118,134,150,166,118,134,150,166,118,134,
    150,166,118,134,150,166,118,134,150,166,118,134,150,166
```

键入程序后，按图 8－31 键控绘图。

绘图完毕，运行程序，将显示图 8－32 的画面，画面上首先在

"Xuan zhi:1,2,3,4"

后有一闪烁光标，等待用户"选指"。假如，我们练习小指，则键入"1"，随后光标移到

"Man su <－－－－－> Kuai su"

之后闪烁，等待调整速度。速度初值定为 100，若欲加快速度则按"→"键，每按一次，速度值减少 10 个单位，若减慢速度则按"←"键，速度调整完毕按"ESC"键开始。这时在两个小指字符键区某一个字符出现闪烁，如"Q"闪烁，则用小指按"Q"键。按 Q 键后，键盘上方左端"DUI:"后记录键对的字符数，然后，又有另一个字符闪烁……。可以一直练习下去。每一个字符闪烁时，键盘上方右端，"Shi Jian:"后不停变换着时间的数值，若该数值为 0 时没有键对，则自动另换一个字符。

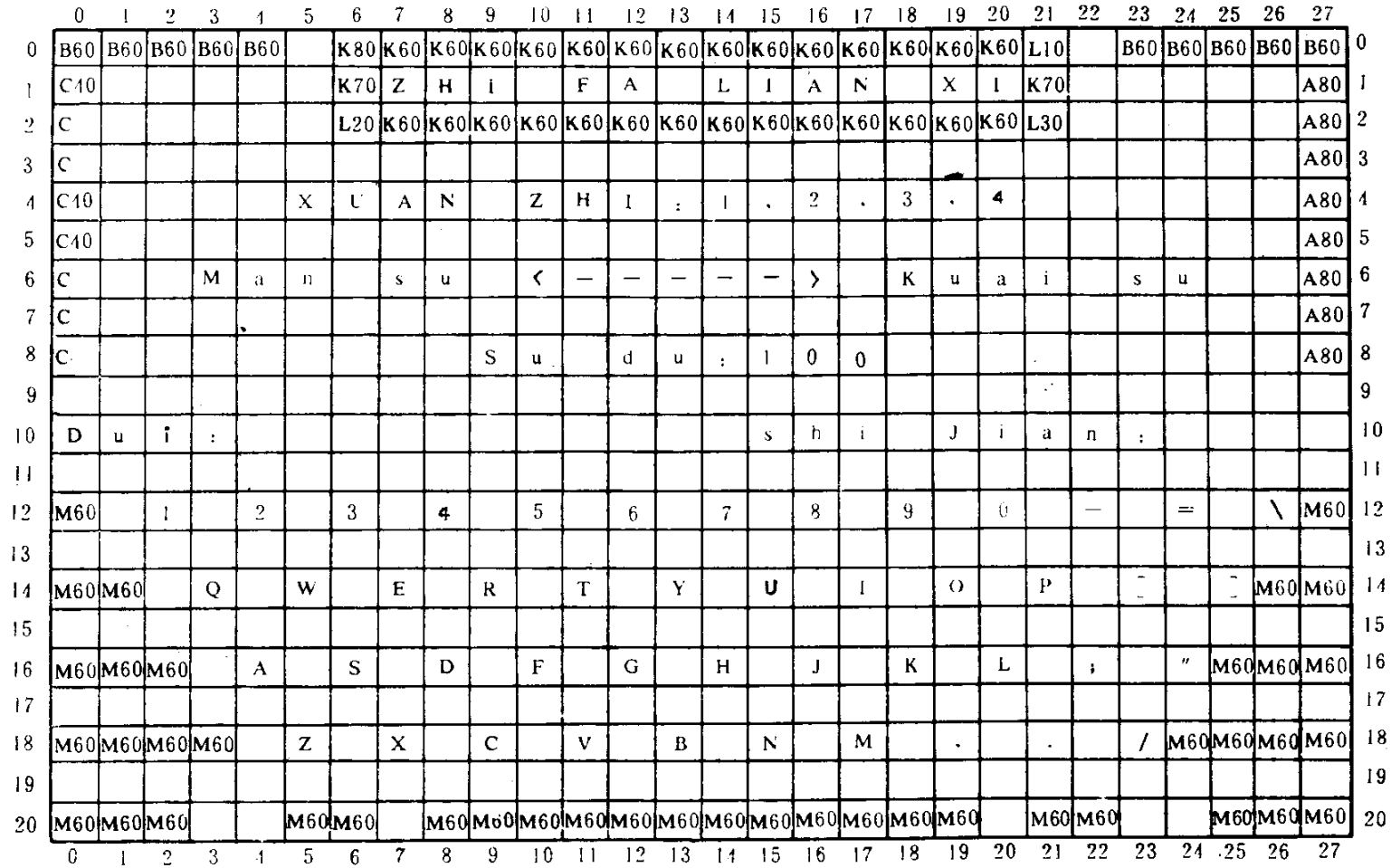


图 8—31

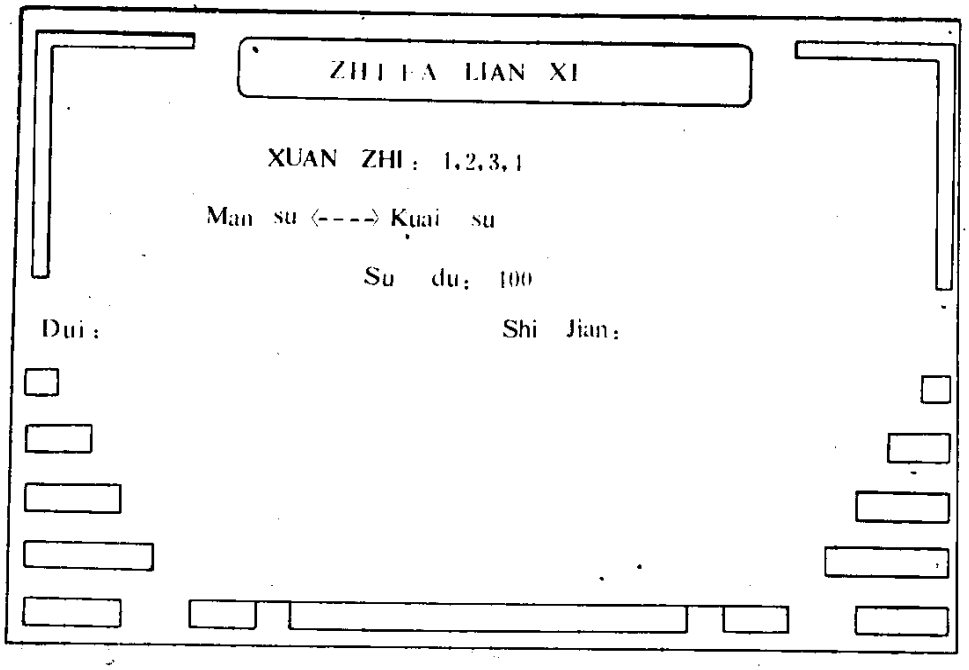


图 8—32

若欲换另一指练习，可按“ESC”键，重新选择。若欲结束练习，按“ESC”后再按“Ctrl＋Z”键即可停机。

程序说明

①10～50 行为初始化赋值程序。其中 20 行定义 B(40)数组装 40 个基本键的 ASCII 码，定义两维数组 A(1,40)装 40 个基本键的位置座标，并对时间赋初值 100。50 行定义 M8 背景图案为 0 号显示 SP，用于字符闪烁。

②60 行转 160～250 行的子程序进行选项，其中 160～170 行进行选指处理；180～220 行为速度选择处理；230 行控制开始(按 ESC 键)；240 行控制结束(按 Ctrl＋Z 键)。

③70～80 行随机产生选定指区的字符。

④90～150 行为操作者输入字符的处理程序。其中 90 行记录时间；100～110 行令字符闪烁；120、140 行为读键盘和键对的处

理;130 行控制返回选项。

⑤270 行为 40 个字符的 ASCII 码;280 行为 40 个字符的显示 X 座标;290 行为 Y 座标。

⑥程序 No. 8－64 仅选定 40 个基本键进行练习,读者欲练习所有键和上档键的字符,只要调整 20、70 和 DATA 语句的有关数据即可,在读懂本程序后,这一调整将很容易实现。

最后再说一句,指法练习是一项很艰苦的工作,希望读者有恒心、有耐力、克服困难,胜利度过这一关。

8.6.4 键盘输入综合练习

进行指法练习后,读者基本掌握了每个指头的击键要领,可以进入综合练习阶段。

在该阶段要重点掌握各上档字符的输入,练习程序见 No. 8－65。

```
5 REM No. 8－65
10 FOR I=0 TO 21:B$=B$+" ":NEXT
20 VIEW:LOCATE 23,10:A$=INKEY$(0):A=VAL(A$)
30 IF A<1 OR A>3 THEN 20
40 ON A GOTO 70,50,60
50 CLS: LIST:PRINT "Press any key to return!":A$=INKEY
   $(0):GOTO 20
60 GOSUB 240:LOCATE 8,6:PRINT "GOOD BYE!!!":LOCATE 0,
   21:END
70 GOSUB 240:H=5 L=0:GOSUB 250
80 FOR I=150 TO 0 STEP －1:LOCATE 0,22:PRINT "Time is" I
   "  "
90 FOR J=1 TO 250
100 A$=INKEY$
110 IF A$=C$ THEN S=S+1:LOCATE 24,1:PRINT S;: GOSUB
   250
```

```
120 NEXT
130 NEXT
140 FOR I=1 TO 10:BEEP:NEXT:CLS
150 PRINT:PRINT "  Time is over!":PRINT
160 PRINT "  In 3 minutes,you typed right"S" keys!":PRINT
170 PRINT "  That's"S/3" chars per min."
180 IF S<250 THEN PRINT:PRINT "  Do make great efforts!"
190 IF S>249 AND S<300 THEN PRINT:PRINT "  Not bad!"
200 IF S>=300 THEN PRINT:PRINT "  Very good!"
210 LOCATE 2,21:PRINT "Try again? (Y/N)" ;: Z$=INKEY
    $(0)
220 IF Z$="Y" THEN 20
230 CLS: GOTO 60
240 FOR I=4 TO 10:LOCATE 2,I:PRINT B$;: NEXT
250 C$=CHR$(RND(58)+33)
260 L=L+1:IF L>26 THEN H=H+1:L=1
270 LOCATE L,H:PRINT C$
280 RETURN
```

输入程序后按图 8－33 进行键控绘图。

绘图后，运行程序，将显示图 8－34 的选项画面。按“1”键，练习开始；按“2”键，列出程序清单；按“3”键结束。

按“1”键后，菜单消失，在下面一栏第一行出现字符，操作者必须在字符出现后立即按相应键，若键对则显示下一个字符，同时在上栏最右端显示键对的字符数，在屏幕下方提示时间的剩余量。练习时间每次三分钟。到时间后，清屏，打印练习成绩和评价。显示的内容及中文说明如下：

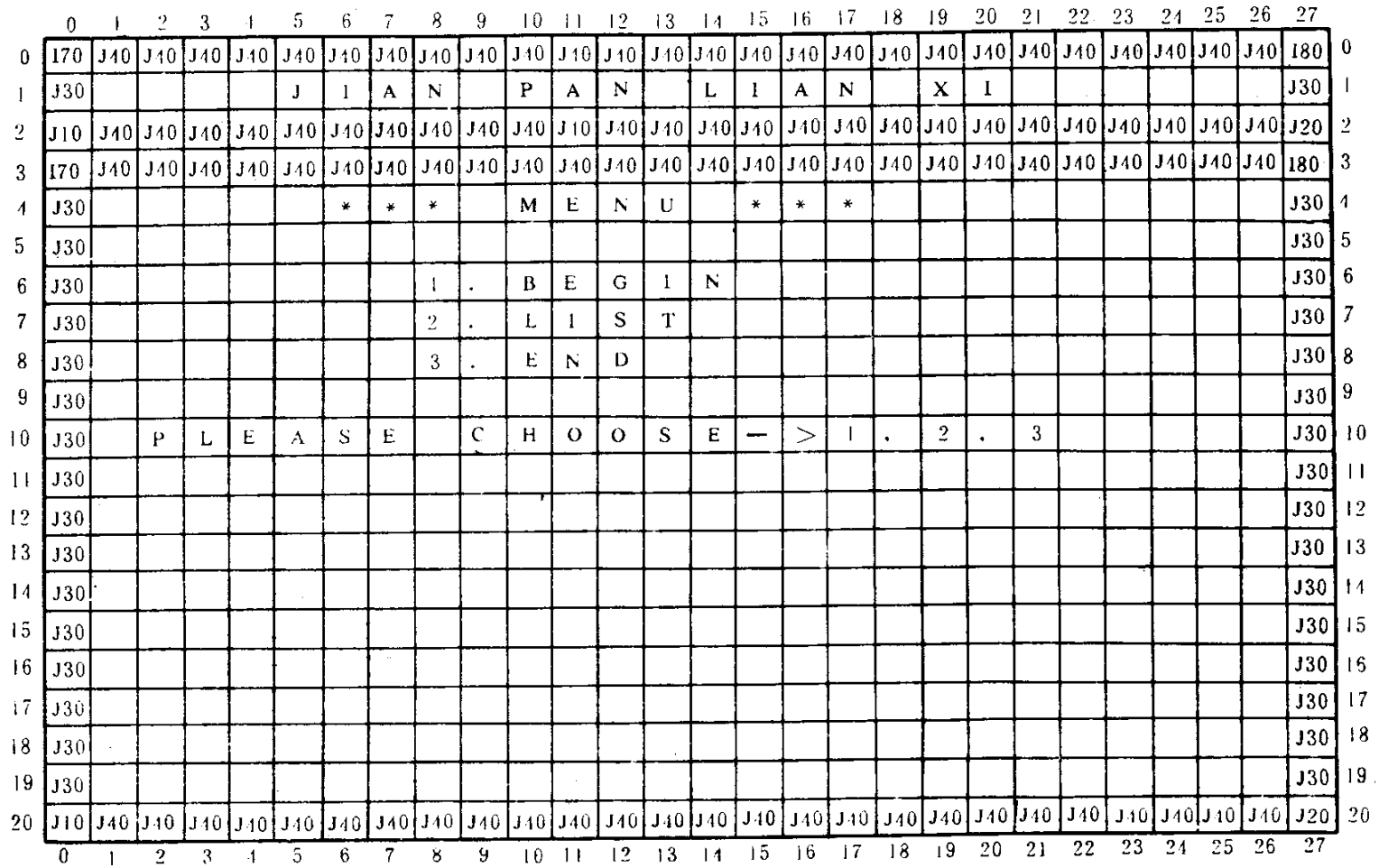

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
0 J70 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J80
1 J30 J A N P A N L I A N X I J30
2 J10 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J10 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J20
3 J70 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J80
4 J30 * * * M E N U * * * J30
5 J30 J30
6 J30 1 . B E G I N J30
7 J30 2 . L I S T J30
8 J30 3 . E N D J30
9 J30 J30
10 J30 P L E A S E C H O O S E — > 1 . 2 . 3 J30
11 J30 J30
12 J30 J30
13 J30 J30
14 J30 J30
15 J30 J30
16 J30 J30
17 J30 J30
18 J30 J30
19 J30 J30
20 J10 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J40 J20
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27

图 8—33

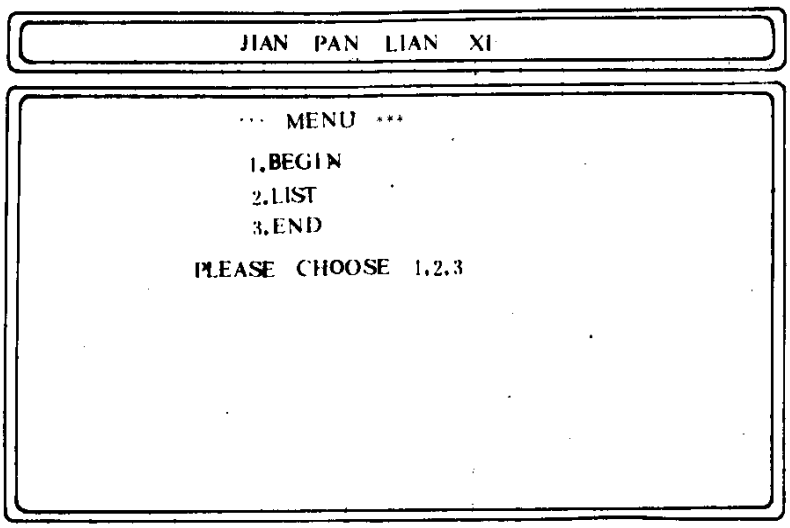


图 8－34

Time is over!　（时间到）

In 3 minutes,you typed right 153 keys

（在 3 分钟内，你共计键对 153 个字符）

That's 51 chars per min　（每分钟 51 个字符）

Do make great efforts　（继续努力）

Not bad!　（不错）

Very good!　（很好）

Try again? (Y/N)　（继续按"Y"键）

若按 Y 键则进行下一次练习；若按其它键则结束运行。

程序说明：

①10～40 行为初始化赋值和选项程序。

②50 行为选项程序。按"1"键练习开始；按"2"键列出程序清单；按"3"键结束运行。

③60 行为结束处理程序，其中调用 240 行子程序进行部分清屏。

④70～130 行完成时间控制，随机产生字符，读取键盘输入，判断键入的字符是否正确，若正确，累计数量，再打印下一个字符

等一系列操作。其中 GOSUB 250 为调用产生、打印字符子程序(250～280 行)。

⑤140～200 行完成练习成绩的评价、评语操作。

⑥210～230 行为继续、结束选项程序。

⑦240 行为部分清屏子程序。

⑧250～280 行为字符产生打印子程序。其中 260 行确定字符的打印位置。

8.6.5 击键速度测试程序

进行键盘的综合练习后,读者已基本熟悉了各字符的键位,可以运行程序 No. 8－66,检查你的击键速度达到什么水平。通过 No. 8－66 的练习可以进一步提高你的击键速度。

程序清单如下:

```
5 REM No. 8－66
10 CLS: DIM A$(28),T$(4):GOSUB 510
20 LOCATE 7,1:GOSUB 380
30 LOCATE 5,3:PRINT "Speed choice"
40 FOR I=0 TO 4:LOCATE 2,5+I:PRINT I "." T$(I):NEXT
50 LOCATE 0,12:PRINT "      Press－－0、1、2、3、4 key";: INPUT
   TT$:TT=VAL(TT$):IF TT<0 OR TT>4 THEN BEEP:GO-
   TO50
60 T=－54*(TT=0)－5*(TT=4):GOSUB 390:CLS
70 LOCATE 7,1:GOSUB 380:PRINT "["T$ "]":D=0:E=0:K=0:
   L=0:P$=" ":PP$=" "
80 LOCATE 2,4:PRINT "You will type with"
90 LOCATE 7,6:PRINT T$(TT)
100 LOCATE 4,21:PRINT "Press any key to begin!";: A$=INKEY
    $(0)
110 LOCATE 0,21:PRINT "                              "
120 FOR I=1 TO 28 A$(I)=CHR$(RND(26)+65):P$=P$+A
```

```
    $(I):NEXT
130 P$=RIGHT$(P$,28):LOCATE 0,8:PRINT P$;
140 FOR I=1 TO 28:LOCATE I-1,9:PRINT "^";
150 S=0
160 A$=INKEY$:A=ASC(A$)
170 IF A$=A$(I) THEN 210
180 IF A>64 AND A$<>A$(I) THEN 220
190 S=S+1:IF S<=T THEN 160
200 K=K+1:BEEP:PAUSE 10:BEEP:PP$=PP$+"!":GOTO 230
210 D=D+1:PLAY "D0C":PP$=PP$+"*":GOTO 230
220 E=E+1:BEEP:PP$=PP$+A$(I)
230 LOCATE I-1,9:PRINT " "
240 L=L+S:NEXT:L=L/28
250 PP$=RIGHT$(PP$,28):LOCATE 0,9:PRINT PP$:GOSUB
    490
260 FOR I=0 TO 26:LOCATE I,10:PRINT "-";:NEXT:PRINT
270 PRINT "   Right :" D"   * *" D*100/28"%"
280 PRINT "   Wrong :" E"   * *" E*100/28"%"
290 PRINT "   Space:" K"   * *"E*100/28"%":PRINT "   Speed
    :" MT
300 PRINT"  Remark:"L$
310 FOR I=0 TO 26:PRINT "—";: NEXT:PRINT
320 IF D>=27 THEN GOSUB 570
330 PRINT "<Space>---CONTINUE":PRINT "<Return>--
    CHOOSE": PRINT "<Esc>-------Quit": BEEP
340 A$=INKEY$(0):IF A$=" " THEN CLS:GOTO 70
350 IF ASC(A$)=13 THEN CLS:GOTO 20
360 IF ASC(A$)=27 THEN END
370 GOTO 340
380 PRINT "<<KEY  TEST>>";:RETURN
390 T=-(T+10)*(TT=0)-54*(TT=1)-21*(TT=2)-5*
```

```
    (TT=3)-(T-1)*(TT=4)
400 IF T<1 THEN T=0
410 T$=MID$(T$(TT),10,4)
420 RETURN
430 IF D<27 AND TT=0 THEN L$="LOW"
440 IF D<27 AND TT>0 THEN L$="DROP":TT=TT-1
450 IF D=27 THEN ;L$="KEEP"
460 IF D>27 AND TT<4 THEN L$="GO UP": TT=TT+1
470 IF D>27 AND TT=4 THEN L$="VERY GOOD"
480 GOTO 390
490 MT=(L>64)-(L<=64 AND L>=54)-2*(L<54 AND L>
    =20)-3*(L<20 AND L>=5)-4*(L<5)
500 GOTO 430
510 T$(0)="LOW1SPEED< 28 per min."
520 T$(1)="LOW2SPEED> 28 per min."
530 T$(2)="MID. SPEED< 56 per min."
540 T$(3)="QUI. SPEED<120 per min."
550 T$(4)="HIGHSPEED>120 per min."
560 RETURN
570 PLAY "O3C1DEFGAB04C3":RETURN
```

运行程序，显示如图 8－35 的画面，画面中的闪烁光标等待用户选择速度。画面的中文意思如下：

《KEY TEST》

Speed choose:

0. LOW1SPEED< 28 per min.

1. LOW2SPEED> 28 per min.

2. MID. SPEED< 56 per min.

3. QUI. SPEED<120 per min.

4. HIGHSPEED>120 per min.

Press－－0,1,2,3,4 key □

图 8－35

速度选择：

0. 低速 1 ＜28 个字符/分钟

1. 低速 2 ＞28 个字符/分钟

2. 中速 ＜56 个字符/分钟

3. 快速 ＜120 个字符/分钟

4. 高速 ＞120 个字符/分钟

按 0、1、2、3、4 键选择速度。

假如我们选择“低速 1”则按“0”键，清屏。显示如图 8－36 的画面。

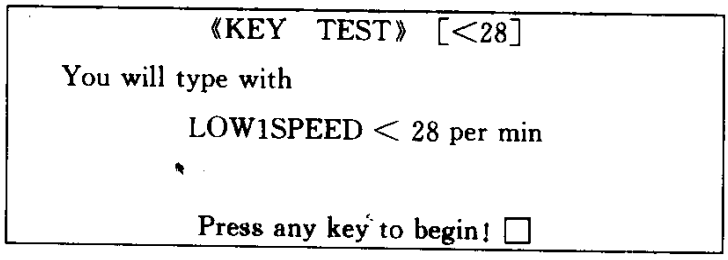


图 8－36

画面表示“你将以每分钟小于 28 个字符的速度击键，按任一键开始”。按任一键后，清除最底一行提示，在屏幕中央随机打印出 28 个字符，第一个字符下面出现“^”击键提示，击键提示符自左向右运动，你必须在移动到该字符下面时击该键，否则无效。当“^”号移动到第 28 个字符以后，本次测验结束。打印测验成绩。假定本次测验出题字符为

“QABPPZWXBCDAYZCCDEBBBTVRSTUV”

则成绩评定画面如图 8－37。

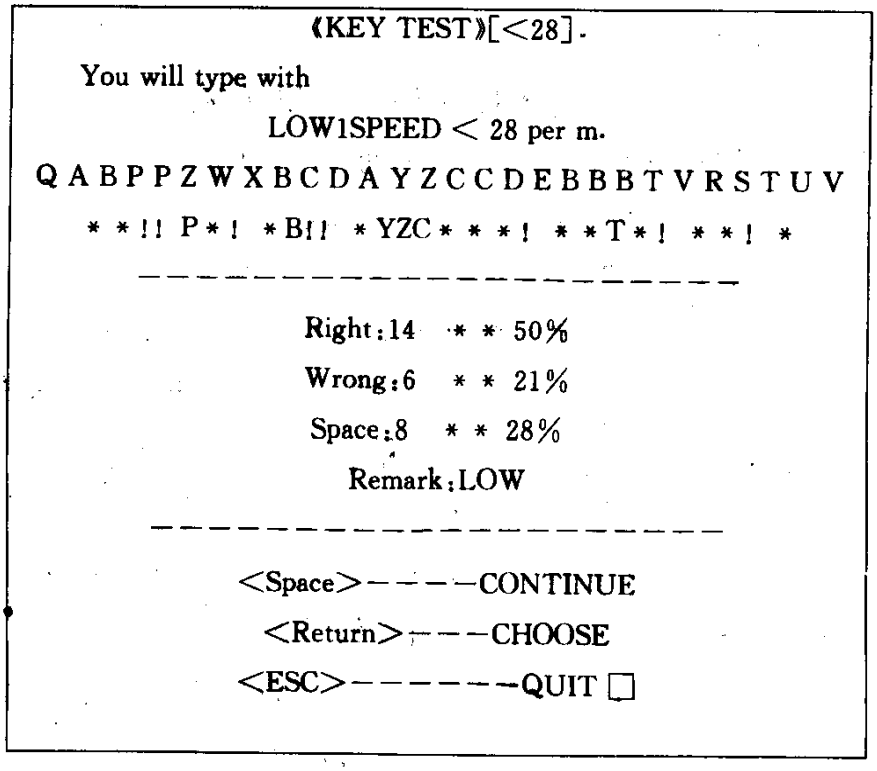


图 8－37

图 8－37 中，字母下面的一行字符中，“＊”表示键对；“!”表示漏键；“字母”表示键[illegible]。

键对 14 个，[illegible]50%

键错 6 个，占 21%

漏键 8 个，占 28%

评价：初级

按空格键：继续。

按回车键：选项。（返 8－35 画面）

按 ESC 键：结束。

根据以上说明，作出需要的选择。

程序说明：

①10 行为初始化赋值程序。其中定义数组 A＄(28)装每次测验的 28 个字母；数组 T＄(4)装 5 种击键速度。调用 510～560 行子程序，对 T＄数组中各变量赋值。

②20 行打印标题。

③30～60 行进行速度选择和确定击键测验的速度。并调用 390～420 行子程序，把击键速度级别赋值给 T $。

④70～100 行打印测验画面图 8－36。

⑤110～130 行随机产生并打印 28 个字母。

⑥140～240 行为击键测验程序。140 行指示应键入的字母；160 行读入键值；170～180 行对击键进行判断处理，若键对转 210 行计数，若键错则转 220 行计数，若漏键则由 200 行计数。

⑦250～320 行为测验成绩评价程序。

⑧330～370 行为测验后选项处理程序。若继续测验按空格键；若选择速度按回车键；若退出测验按“ESC”键。

⑨430～480 行为评价处理程序。电脑将根据你的测验成绩做出如下的处理和评价：

LOW：初级。刻苦努力，定能成功。

DROP：降级。不要泄气，争取进步。

KEEP：保持。继续努力，成功在即。

GO UP：升级。祝你成功，再接再励。

VERY GOOD：优秀。发扬成绩，更进一步。

⑩程序 No. 8－66 只测验了 26 个字母。如欲测验所有字符的输入速度，可把 120 行中的 A $ (I)的赋值语句改为：

```
A(I)=CHR $ (RND(58)+33)
```

本节共介绍了四个键盘练习程序。这四个程序由易到难，循序渐进，各有侧重，各有特点。读者只要按四个程序的顺序逐个练习，定能达到专业水平。

第九章　趣味程序的编写技巧

所谓趣味程序是指带有一定的趣味性的实用程序。也就是说，这类程序既有实用性，又有趣味性。实用性使人们使用电脑能够处理一些生活中的问题；趣味性可以提高人们学习微机的兴趣。从而实现从娱乐中领会编写程序的方法、思路；在实践中掌握把具体问题抽象化、建立数学模型、编程解决的技术窍门。

9.1　电脑猜谜

9.1.1　电脑能猜到你喜欢的日子

这是一个电脑猜数游戏。程序清单如下：

```
5 REM No. 9—1
10 T=1 : G=0
20 FOR A=1 TO 5 : CLS : IF A>1 THEN T=T * 2
30 LOCATE 9,1 : PRINT "* * *" A"* * *"
40 V=6 : H=5
50 FOR B=T TO 31
60 BB=B : FOR C=1 TO A : D=BB/2 : E=BB-D * 2 : BB=D :
   NEXT
70 IF E=1 THEN LOCATE H,V : PRINT B : H=H+5 : IF H>20
   THEN H=5 : V=V+3
75 NEXT
80 LOCATE 0,20 : PRINT "DO YOU LIKE? (1/0)"; : F$=INKEY
   $(0) : F=VAL(F$) : G=G+T * F : NEXT
```

```
90 CLS : LOCATE 5,10 : PRINT "YOUR FAVOR IS  "G
100 LOCATE 3,24 : PRINT "TRY AGAIN?"; : A $ =INKEY $ (0)
    : IF A $ <>"N" THEN RUN
110 CLS : LOCATE 8,8 : PRINT "GOODBYE!!" : LOCATE 0,22 :
    END
RUN↙打印第一张数据表
        * * *  1  * * *
        1    3    5    7
        9    11   13   15
        17   19   21   23
        25   27   29   31
DO YOU LIKE? (1/0)□
```

在第一张表中，若有你喜欢的日子则按数字键“1”；若无你喜欢的日子则按“0”键。当按过“1”或“0”键后则清屏，打印第二张表供选择。选择五张表后，则打印出你喜欢的日子。假如在第一张表中我们选择“21”，则键入数字“1”，随后显示第二张表：

```
        * * *  2  * * *
        2    3    6    7
        10   11   14   15
        18   19   22   23
        26   27   30   31
```

第二张表中无“21”，则键入“0”，接着显示第三张表：

```
        * * *  3  * * *
        4    5    6    7
        12   13   14   15
        20   21   22   23
        28   29   30   31
```

第三张表中有：“21”，键入‘1’。打印第四张表。

```
        * * *  4  * * *
        8    9    10   11
```

12　13　14　15
24　25　26　27
28　29　30　31

第四张表中无“21”，键入“0”后则显示第五张表。

* * *　5　* * *
16　17　18　19
20　21　22　23
24　25　26　27
28　29　30　31

第五张表中有“21”则键入“1”。清屏，显示

YOUR FAVOR IS 21
TRY AGAIN? □

意思是：你喜欢的数是 21。再玩一次吗？若继续游戏则键入除“N”键外的任一键，则又从第一张表开始进行人机对话。若按“N”键则清屏，打印

GOODBYE!!

停机

程序说明：

①50～75 行是程序的主体，它完成打印一张 4×4 的数据表。其中由 50、60 两行产生数字，70 行控制数字的打印座标。

②20、30、80 三行语句的作用是控制打印五张表，其中 80 行完成人机对话和计算游戏者选择的日期。

③90 行打印结果

④100、110 两行为结束处理程序。

程序 No. 9—1 的运行速度较慢。下面再列出一个运行速度较快的程序清单。见 No. 9—2

```
5 REM No. 9—2
10 T=1 : G=0
20 FOR A=1 TO 5 : CLS : IF A>1 THEN T=T * 2
```

```
30 LOCATE 9,2 : PRINT " * * * "A" * * * " : V=6 : H=0
40 FOR B=T TO 31 STEP T * 2 : FOR C=B TO B+T-1
50 H=H+5 : IF H>20 THEN H=5 : V=V+3
60 LOCATE H,V : PRINT C : NEXT : NEXT
70 LOCATE 0,20 : PRINT "DO YOU LIKE? (1/0)";F $ =INKEY
   $ (0) : F=VAL(F $ ) : G=G+T * F : NEXT
80 CLS : LOCATE5,10 : PRINT "YOUR FAVOR IS  "G
90 LOCATE 3,20 : PRINT "TRY AGAIN?"; : A $ =INKEY(0) : IF
   A $ <>"N" THEN RUN
100 CLS : LOCATE 8,8 : PRINT "GOODBYE!!" : LOCATE 0,22 :
    END
```

两程序的编程思路相同，运行结果相同，但 No. 9—2 比 No. 9—1 的运行速度要快的多。运行速度加快的原因是 40 行的循环步长加大，其编程原理留给读者分析。

9.1.2 电脑猜属相

中国人之间询问年龄，往往不直接回答自己的岁数，而告之以属相。知道了属相也就等于知道了年龄。你知道你的属相吗？请告诉电脑你的出生年份，它便能很快地猜出你的属相。

另外，中国古人使用天干地支记年份，如"甲午"年、"庚申"年、"丙寅"年等。如果你正在研究历史或阅读古籍，想知道公元前某年的干支、生肖，也可以向电脑查询。你定能得到满意的答复。源程序如下：

```
5 REM No. 9—3
10 CLS
20 DIM M $ (10),N $ (12)
30 FOR M=0 TO 9 : READ M $ (M) : NEXT
40 FOR N=0 TO 11 : READ N $ (N) : NEXT
50 INPUT "INPUT nian hao=";Y : PRINT
60 IF Y=0 THEN PRINT "Wu0nian! INPUT AGAIN!" : GOTO 50
70 M=Y-ABS(Y)/10 * 10 * SGN(Y)
```

```
80 N=Y-ABS(Y)/12*12*12*SGN(Y)
90 IF Y<0 THEN M=M+11:M=M-M/10*10:N=N+13:N=
   N-N/12*12:PRINT "Gong yuan qian":GOTO 110
100 PRINT "Gong yuan";
110 PRINT "  "ABS(Y);"nian:":PRINT
120 PRINT "              "M$(M)"    "N$(N):PRINT:
    PRINT:PRINT
130 PRINT "Jixu ma? (Y/N)";:A$=INKEY$(0)
140 IF A$="Y" THEN CLS:GOTO 50
150 END
160 DATA Geng,Xin,Ren,Gui,Jia,Yi,Bing,Ding,Wu,Ji
170 DATA Shen(Hou),You(Ji),Xu(Gou),Hai(Zhu),Zi(Shu),Chou
    (Niu),Yin(Hu),Mao(Tu),Chen(Long),Si(She),Wu(Ma),Wei
    (Yang)
```

程序说明：

为便于不懂英语的朋友使用，干支生肖及输入提示均采用汉语拼音。下面对160、170两行的数据说明如下：(用中文表示)

```
160 DATA 庚,辛,壬,癸,甲,乙,丙,丁,戊,己
170 DATA 申(猴),酉(鸡),戌(狗),亥(猪),子(鼠),丑(牛),寅(虎),卯
    (兔),辰(龙),巳(蛇),午(马),未(羊)
```

读者可对照识别。

1. 10～40行为初始化程序。其中30行把160行的天干数据装入数组M$中，40行把170行的地支及属相数据装入数组N$中。

2. 50～60行为键盘输入语句。公元前的年号用负数输入，如“公元451年”则输入：—451”。

3. 70～120行程序完成干支、属相的计算和输出。

4. 130～150行为结束处理程序。

运行程序，显示如下内容：

INPUT nian hao=?

提示用户:输入年号。若输入 1992 年,则打印

Gong yuan 1992 nian:

Ren Shen(Hou)

意思是:公元 1992 年:壬申(猴)

若继续查询则按"Y"键;否则按除"Y"外的任一键,停机。

9.1.3 电脑测星期

我们知道 1991 年的国庆节是星期二。1992 年的国庆节是星期四,也可以推出 1993 年的国庆节是星期五……。那么 2000 年的国庆节是星期几呢? 3000 年的呢? 1949 年的开国大典日是星期几呢? 芦沟桥事变呢? 八一南昌起义日呢? 这些,推算出来要颇费一番脑筋。若运行 No. 9—4 的程序,电脑可以迅速回答你想要了解的任一天是星期几。

```
5 REM No. 9—4
10 CLS : DIM M(12),A $ (6)
20 DATA 31,28,31,30,31,30,31,31,30,31,30,31
30 DATA SUNDAY(7),MONDAY(1),TUESDAY(2),WEDNES-
   DAY(3),THURSASY(4),FRIDAY(5),SATURDAY(6)
40 FOR I=1 TO 12 : READ M(I) : NEXT
50 FOR J=0 TO 6 : READ A $ (J) : NEXT
60 INPUT "INPUT DATA : YEAR,MONTH,DAY=";Y,M,D
70 IF M=1 THEN 100
80 FOR I=1 TO M-1 : C=C+M(I) : NEXT
90 IF (Y MOD 4=0)AND (Y MOD 100<>0) OR (Y MOD 100=0 )
   AND (Y MOD 400=0) THEN C=C+1
100 YY=Y-1 : S=YY+YY/4-YY/100+YY/400+C+D
110 S=S MOD 7
120 PRINT Y". " M". " D". IS";A $ (S) : PRINT
130 PRINT "PRESS ANY KEY TO BEGIN!" : B $ =INKEY $ (0) :
    IF B $ ="N" THEN END
```

```
140 C=0 :GOTO 60
150 END
```

程序说明：

①10～50 行为初始化程序。40 行把 12 个月的天数装入 M 数组；50 行把星期一～星期日的数据装入 A $ 数组中。

②60 行是人机对话程序，用户在该行输入年、月、日的数据。

③80 行累计计算年份的从一月至计算月份前一月的天数。

④90 行是对闰年多加一天的处理。

⑤100～120 行计算输出结果。

运行程序，屏幕显示

```
INPUT DATA:YEAR,MONTH,DAY=? □
```

依次输入“年、月、日”。如输入 1992,10,,1↙则打印

```
1992. 10. 1. IS THURSDAY(4)
PRESS ANY KEY TO BEGIN! □
```

按除“N”键外的任一键重新开始。

现在我们输入“1938,9,18”则打印

```
1938. 9. 18 IS WEDNESDAY(3)
```

我们若分别输入“1949,10,1”,“2000,10,1”,“3000,10,1”。则分别打印

```
1949. 10. 1 IS SATURDAY(6)
2000. 10. 1 IS SUNDAY(7)
3000. 10. 1 IS THURSDAY(4)
```

9.1.4 电脑猜血型

医学研究表明子、女的血型与父、母的血型密切相关。法庭上也依父母、子女血型为根据判定子女属于亲生还是抱养。你知道自己的血型吗？运行程序 No. 9—5a，输入父母的血型就可以知道自己的可能血型。

```
5 REM No. 9—5a
10 CLS:DIM A $ (4)
```

```
20 DATA A,B,AB,O
30 INPUT "father's=";F$ : INPUT "mother's=";M$ : PRINT
   "POSSIBLY=";
40 FOR I=1 TO 4: READ A$(I)
50 IF F$=A$(I) THEN U=I
60 IF M$=A$(I) THEN V=I
70 NEXT : IF U>V THEN SWAP U,V
80 ON U GOTO 90,110,120,140
90 IF V=1 OR V=4 THEN PRINT "A,O" : END
100 IF V=2 THEN PRINT "A,B,AB,O" : END
110 IF V=2 OR V=4 THEN PRINT "B,O" : END
120 IF V=1 OR V=2 OR V=3 THEN PRINT "A,B,AB" : END
130 IF V=4 THEN PRINT "A,B" : END
140 IF V=4 THEN PRINT "O" : END
150 PRINT "NONE!" : END
RUN↙屏幕显示
father's=? □        (要求输入父亲的血型)
```

假设父亲的血型为 A 型，则键入"A"↙显示

```
mather's=? □      (要求输入母亲的血型)
```

假设母亲的血型为 B 型，则键入"B"↙打印

```
POSSIBLY=A,B,AB,O
```

表示子女的可能血型为"A"型、"B"型、"AB"型或"O"型。

若输入父、母的血型分别为 A 型、O 型则打印：

```
POSSIBLY=A,O
```

表示子女的可能血型为"A"型或"O"型。

程序说明：

①30 行的键盘输入语句，要求依次输入父、母亲的血型。

②40～70 行的作用是根据父、母的血型计算出 U、V 的值。并使 U〈V。

③80～150 行完成对各种可能血型的判断，各语句的设置以

表 9—19(父、母、子、女血型对照表)为依据。读者可自行分析。

表 9—1a　父母子女血型对照表

父、母血型		子、女可能血型
O,A	A	O,A
B	A	O,AB,A,B
AB,A,B	AB	A,B,AB
O,B	B	O,B
O	AB	A,B
O	O	O

程序 No. 9—5a 只能计算推测出子女的血型,若已知子女和双亲中一方的血型,欲求父亲或者母亲的可能血型应怎么办呢?显然程序 No. 9—5a 没有这一功能,因此,我们得另辟蹊径。

归纳表 9—1a 中子女血型的各种组合,可以列出表 9—1b。表 9—1b 表明,子女的任何一种血型都可看作是两个因子的组合,这两个因子分别来自父亲和母亲,所以只要将父母血型中的两个因子进行组合,其结果就是子女的可能血型。反之,将这个过程逆推,就能求得双亲中未知一方的可能血型。程序见 No. 9—5b。

表 9—1b　血型组成对照表

血　型	组　成
A	AO 或 AA
B	BO 或 BB
O	OO
AB	AB

```
5 REM No. 9—5b
10 CLS : DIM M$(4),N$(7),B$(3),P$(7)
20 FOR I=1 TO 4 : READ M$(I) : NEXT
30 FOR I=1 TO 7 : READ N$(I) : NEXT
40 DATA O,A,B,AB
50 DATA Fuqin,Muqin,Zinu,Xuexing,Keneng,"  ",Bucunzai
60 PRINT "Xuexing Weizhi an<N> jian" : PRINT
70 FOR I=1 TO 3
```

```
80 PRINT N$(I);"  "N$(4); : INPUT B$(I)
90 IF B$(I)="N" THEN W=I:N$(6)=N$(I)
100 NEXT : PRINT
110 PRINT N$(6) "  "N$(5)"  "N$(4)":";
120 IF W=2 THEN U=2 : V=1 : GOTO 140
130 U=1 : V=2 : IF W=3 THEN GOSUB 170 : END
140 FOR M=1 TO 4 : B$(U)=M$(M) : GOSUB 170 : NEXT
150 IF P=0 THEN PRINT N$(7)
160 END
170 IF LEN(B$(V))=1 THEN B$(V)=B$(V)+"0"
180 IF LEN (B$(U))=1 THEN B$(U)=B$(U)+"0"
190 FOR I=1 TO 2
200 K$=MID$(B$(U),I,1)
210 FOR J=1 TO 2
220 L$=MID$(B$(V),J,1) : Z$=K$+L$
230 IF ASC(K$)>ABS(L$) THEN Z$=L$+K$
240 IF RIGHT$ (Z$,1)=LEFT$(Z$,1) OR RIGHT$(Z$,1)=
    "0" THEN Z$=LEFT$(Z$,1)
250 IF W=3 THEN 280
260 IF Z$<>B$(3) THEN 320
270 Z$=M$(M)
280 FOR N=0 TO P
290 IF Z$=P$(N) THEN N=P+1
300 NEXT : IF N<>P+1 THEN 320
310 P=P+1 : P$(P)=Z$ : PRINT Z$",";
320 NEXT : NEXT : RETURN
```

程序说明：

①各拼音说明：

Fuqin	父亲，	Muqin	母亲
Zinu	子女，	Xuexing	血型

Keneng　可能，　　Bucunzai　不存在

Xuexing weizhi an<N>jian

血型未知按 N 键

②10～50 为初始化程序，定义 M $(4)数组装血型；N $(7)数组装说明；B $(3)数组装输入的父、母、子女血型；P $(7)数组装程序运行中的有关血型。

③60～100 行为人机对话程序。由用户输入父、母子女的血型（包括欲查询的血型）。

④120～160 行对查子女血型或双亲中一方血型进行分别处理。其中 120、140～160 行为查双亲血型的处理；130 行是查子女血型的处理。

⑤170～180 行补齐已知血型中省写的因子，如“A”补齐为“AO”，“B”补齐为“BO”，“O”补为“OO”。这样做的目的是便于后续程序的处理。

⑥190～230 行对两个已知血型中的因子进行排列组合，并保证实际相同的组合中因子的排列顺序一致。例如“AO”和‘OA”同为“A”型血，处理后都按“AO”顺序排列。

⑦240 行完成把组合结果转换成我们习惯的写法。例如：“AO”转换为“A”。

⑧280～310 完成将所有的血型组合不重复地打印出来。

现在，我们使用 No. 9—5b 求血型。

①求子女的血型

运行程序，显示

Xuexing weizhi an〈N〉jian

Fuqin Xuexing?　若键入“A”↙

Muqin Xuexing?　若键入“B”↙

Zinu Xuexing?　键入“N”↙

Zinu Keneng Xuexing:AB,A,B,O

RUN↙

Fuqin Xuexing?　若键入“B”↙

Muqin Xuexing?　若键入“AB”↙

Zinu Xuexing?　键入“N”↙

Zinu Keneng Xuexing:AB,B,A

②查父亲的血型

RUN↙

Fuqin xuexing?　键入“N”↙

Muqin Xuexing?　若键入“A”↙

Zinu Xuexing?　若键入“B”↙

Fuqin Keneng Xuexing:B,AB

③查母亲的血型：

RUN↙

Fuqin Xuexing?　若键入“B”↙

Muqin Xuexing?　键入“N”↙

Zinu Xuexing?　若键入“O”↙

Muqin Keneng Xuexing:O,A,B

9.2　数字螺旋方阵

所谓螺旋方阵就是一组从一开始的自然数，从小到大螺旋排列成一个几 n 阶方阵。如 3 阶螺旋方阵有下列形式，如图 9—1。

1　8　7

2　9　6

3　4　5

图 9—1　左上逆旋 3 阶方阵

图 9—1 称为“左上逆旋 3 阶方阵”。若为图 9—2 的形式则称为“右上顺旋 5 阶方阵”。

13　14　15　16　1

12　23　24　17　2

11　22　25　18　3

10 21 20 19 4

9 8 7 6 5

图 9—2 右上顺旋 5 阶方阵

不难算出，外向内旋的数字螺旋方阵共有“顺、逆”向两类八种形式。（见表 9—2）

表 9—2 外向内旋方阵表

顺 旋	逆 旋
右 上	右 上
右 下	右 下
左 上	左 上
左 下	左 下

同样由内向外旋方阵也有“顺、逆”向两类八种形式。因此螺旋方阵共有 16 种形式。以下介绍打印各种螺旋方阵的编程方法技巧。

9.2.1 螺旋方阵的实现

为简化叙述，我们以“左上顺旋 3 阶方阵为例进行讨论。

方阵形式如图 9—3

1 2 3

8 9 4

7 6 5

图 9—3 左上顺旋 3 阶方阵

我们把方阵看成一个平面，建立直角座标系，座标原点选在左上角（见图 9—4）。

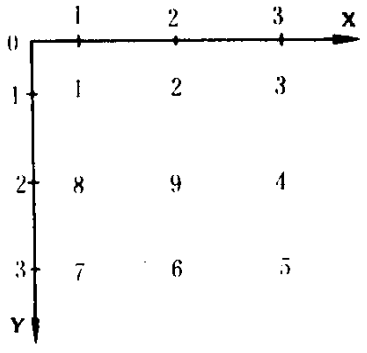


图 9—4 直角座标系中的方阵

由图 9—4 可见，组成方阵的每一个元素的位置对应一组座标。如“1”对应(1,1)、“5”对应(3,3)、“7”对应(1,3)等。由此引出程序的设计思路：首先定义两维数组，然后把方阵中的各元素装进各下标变量中，最后按下标变量的下标为打印座标打印其中的内容。根据编程思路，我们定义两维数组 A(3,3)，令

A(1,1)=1　A(2,1)=2　A(3,1)=3

A(1,2)=8　A(2,2)=9　A(3,2)=4

A(1,3)=7　A(2,3)=6　A(3,3)=5

然后设置双循环语句。

FOR I=1 TO 3:FOR J=1 TO 3:PRINT A(I,J);:NEXT:NEXT

就可打印出方阵。显而易见，编程的关键是如何把方阵各元素装进对应的下标变量中。

对于各元素的装入过程，我们构思如下：首先，使第二下标 Y=1，令第一下标 X 从 1 变化到 3，则可把“1”、“2”、3”分别装入 A(1,1)、A(2,1)、A(3,1)中，即有

A(1,1)=1、(2,1)=2、A(3,1)=3

然后令 X=3 不变，令 Y 从 2 变到 3，把“4”、“5”装入 A(3,2)、A(3,3)中，即有

A(3,2)=4、A(3,3)=5

再令 Y=3 不变，令 X 从 2 变到 1 则装入“6”，“7”于 A(2,3)、A(1,3)中，有

A(2,3)=6、A(1,3)=7

再令 X=1 不变，令 Y=2，把 8 装入 A(1,2)中，则有

A(1,2)=8

最后，令 Y=2 不变，令 X=2，把“9”装入 A(2,2)中，有

A(2,2)=9

至此整个工作结束。现在，如果使用双循环语句，定能打印出图 9—3 的方阵。

上述的装入过程可用下列程序实现：

```
20 DIM A(3,3),B(2)
30 B(2)=1 : C=1 : T=1 : H=3
50 FOR I=1 TO 2
60 FOR J=1 TO H
70 B(I)=B(I)+T
80 A(B(1),B(2))=C : C=C+1
90 NEXT
100 IFI=1 THEN H=H-1
110 IF H=0 THEN (转打印方阵程序)
120 NEXT
130 T=-T : GOTO 50
```

程序中引入了一维数组 B(2)分别表示 A(3,3)数组的双下标，B(1)表示 X 座标，B(2)表示 Y 座标；C 代表方阵中的元素；H 控制内循环的终值，及时地令 X 不变或 Y 不变，并令 H=0 时为下标变量赋值结束标志；T 为 B(I)下标的增量，取值为“1”或“-1”。装入程序的工作过程与我们构思的装入过程完全一致。读者可对照分析。不多赘述。

为使方阵中各元素之间在打印时拉开一定的距离，我们对打印座标进行如下设计：

```
LOCATE 3*I-2 ,3*J-2
```

这样，当 I=1、J=1 时，下标变量 A(1,1)的打印座标为(1,1)；当 I=1、J=2 时，A(1,2)的打印座标为(1,4)；当 I=2、J=2，A(2,2)的打印座标为(4,4)。可见，各元素的上、下、左、右之间均隔开 2 个字符的距离。

把以上程序合并，就得到完整的“左上顺旋方阵“通用程序，见 No. 9—6。

```
5 REM No. 9—6
10 CLS:INPUT:INPUT "3～9=";N
20 DIM A(N,N),B(2)
```

```
30 B(2)=1:C=1:T=1 : H=N
40 P=3 : IF N>7 THEN P=2
50 FOR I=1 TO 2 : FOR J=1 TO H
60 B(I)=B(I)+T : A(B(1),B(2))=C : C=C+1 : NEXT
70 IF I=1 THEN H=H-1
80 IF H=0 THEN 110
90 NEXT
100 T=-T : GOTO 50
110 FOR I=1 TO N : FOR J=1 TO N
120 LOCATE 3*I-2,P*J-P : PRINT A(I,J)
130 NEXT : NEXT
```

程序中 40 行的作用是当 N 大于 7 时，使方阵的纵向间隔减小，以保证元素打印的纵向座标不溢出。

通过分析程序可知，若把程序 No. 9—6 中的 A(B(1),B(2))=C 改为 A(B(2),B(1))=C 则打印出“左上逆旋方阵”，请读者验证。

我们知道，顺、逆方阵各有 4 个，以上我们仅编出了左上顺，逆旋方阵的程序，那么对左下、右上、右下 6 个顺、逆旋方阵应如何编程呢？显然，要对 No. 9—6 中的循环赋值程序进行改造，而且要有三种形式，方能打印 6 种方阵。改造的工作量较大。能否仅对程序稍微改动就可打印出各种方阵呢？这要进行更深一步的探讨。

9.2.2 螺旋方阵的编程技巧

在程序 No. 9—6 中，是对各下标变量先计算赋值，再输出打印，程序运行中要使用 N 次双重循环，执行速度较慢。现在，我们讨论一种边计算边打印的编程方法。

1. 左上顺旋方阵

下面分析讨论一个五阶左上顺旋方阵，见图 9—5。

```
 1   2   3   4   5
16  17  18  19   6
```

15　24　25　20　7

14　23　22　21　8

13　12　11　10　9

图 9—5　左上顺旋五阶方阵

在图 9—4 中，我们把座标原点选在方阵的左上角，现在讨论把座标原点移到方阵中间情况，见图 9—6。

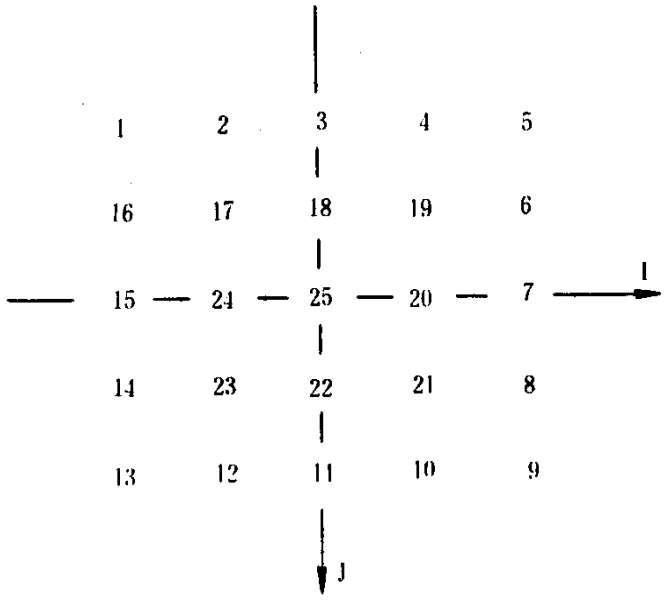


图 9—6　座标系中的五阶左上顺旋方阵

对于图 9—6，若定义数组 A(5,5)后，则有

A(0,0)=25，　A(−2,−2)=1

A(2,−2)=5，　A(2,2)=9

A(−2,2)=13……

图 9—6 中，I、J 的取值均为−2、−1、0、1、2. 。若抽象为数学模型则有：

设方阵阶数为 N，m=(n−1)/2

行的取值为：I=−m，−m+1，−m+2……m−1，m

列的取值为：J=−m，−m+1，−m+2……m−1，m

对于左上顺旋方阵，通过计算，可把它分为 A、B1、B2 三个区域(见图 9—7)。由于 B1、B2 两区域性质相同，可合并为一个区称为 B

区。根据每区各元素的变化规律，我们可归纳写出每区元素的通项表达式：

$$\begin{cases} \text{A 区：} N^2-(4J^2-2J+|I+J|) \quad |I|\leqslant|J| & (1) \\ \text{B 区：} N^2-(4I^2+2I-|I+J|) \quad |I|>|J| & (2) \\ I=-M,-M+1,\cdots\cdots,M-1,M & (3) \\ J=M,M-1,\cdots\cdots,-M+1,-M & (4) \end{cases}$$

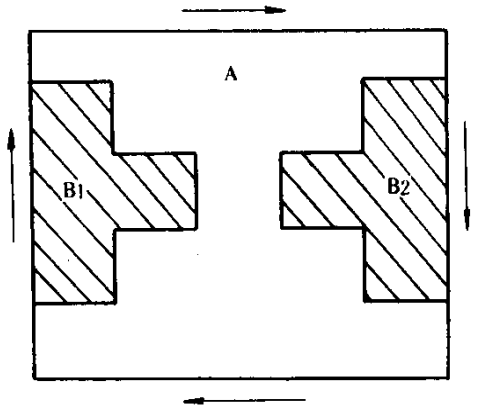


图 9—7　左上顺旋阵分区图

关于分区情况，具体到图 9—6 则有图 9—8 的形式。

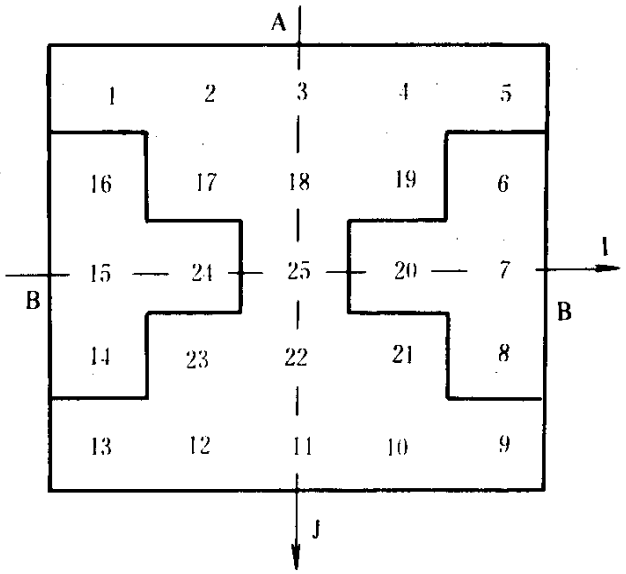


图 9—8

图 9—8 中各区元素的通项表达示为：

$$\text{A 区：} 25-(4J^2-2J+|I+J|) \quad |I|\leqslant|J| \qquad (1)$$

B区：$25-(4I^2+2I-|I+J|)$ $|I|>|J|$ (2)

现在我们验证一下两通项表达示的正确性。图9—8中A区元素“5”的座标为(2，－2)，把

$$\begin{cases} I=2 \\ J=-2 \end{cases}$$

代入(1)式则有

$$\begin{aligned} &25-[4\times(-2)\times(-2)-2\times(-2)+|2+(-2)|] \\ =&25-[16+4+0] \\ =&25-20 \\ =&5 \end{aligned}$$

A区元素“11”的座标为(0，2)，代入(1)式有

$$\begin{aligned} &25-[4\times 2^2-2\times 2+|0+2|] \\ =&25-14 \\ =&11 \end{aligned}$$

可见A区的通项式是正确的，读者可多代入几组值验证两通项式的正确性。

有了各元素的通项表达式，我们就可以根据I、J的取值，很方便地计算出各座标应打印的数值，从而达到边计算边打印的目的。据此，我们编制程序如No. 9—7。

```
5 REM No. 9—7
10 CLS : INPUT "INPUT 3～9,";N : CLS
20 M=(N－1)/2 : P=3 : IF N>7 THEN P=2
30 FOR I=－M TO M
40 FOR J=－M TO M
50 IF ABS(I)>ABS(J) THEN GOTO 70
60 K=N * N－(4 * J * J－2 * J+ABS(I+J)) : GOTO 80
70 K=N * N－(4 * I * I+2 * I－ABS(I+J))
80 LOCATE 3 * (N/2+I),P * (J+N/2) : PRINT K
90 NEXT : NEXT
100 END
```

运行程序，键入“5”，可立即打印出如图 9—5 的左上顺旋 5 阶数字方阵。

我们知道，F BASIC 的运算符中有关系运算符“＞”和逻辑运算符“NOT”，它们的功能作用如下：。

对于“＞”有如下关系

$$X=I>J\begin{cases}\text{若 } I>J & \text{则 } X=-1\\ \text{若 } I=J & \text{则 } X=0\\ \text{若 } I<J & \text{则 } X=0\end{cases}$$

即有

$$I>J=\begin{cases}-1 & \text{当 } I>J \text{ 时}\\ 0 & \text{当 } I\leqslant J \text{ 时}\end{cases}$$

对于“NOT”则有

$$\text{NOT } X=\begin{cases}0 & \text{当 } X=-1 \text{ 时}\\ -1 & \text{当 } X=0 \text{ 时}\\ -2 & \text{当 } X=1 \text{ 时}\end{cases}$$

使用“＞”和“NOT”可使程序 No. 9—7 简化，简化程序见 No. 9—8

```
5 REM No. 9—8
10 CLS : INPUT "INPUT N=3、5、7、9",N
20 M=(N-1)/2 : P=3 : IF N>7 THEN P=2
30 FOR I=-M TO M : FOR J=-M TO M
40 X=ABS(I)>ABS(J)
50 K=N * N+(4 * I * I+2 * I-ABS(I+J)) * X+(4 * J * J-2 * J+
   ABS(I+J)) * NOT X
60 LOCATE 3 * (N/2+I),P * (N/2+J) : PRINT K
70 NEXT : NEXT
80 END
```

通过讨论，找到了编制所有螺旋方阵程序的有效方法。即：对不同的螺旋方阵，只要找出方阵元素的通项表达式，然后修改程序 No. 9—8 中的 40、50 两行语句，就可求出其它 15 种螺旋方阵。

2. 右上顺旋方阵

右上顺旋方阵的 A、B 区划分如图 9—9。

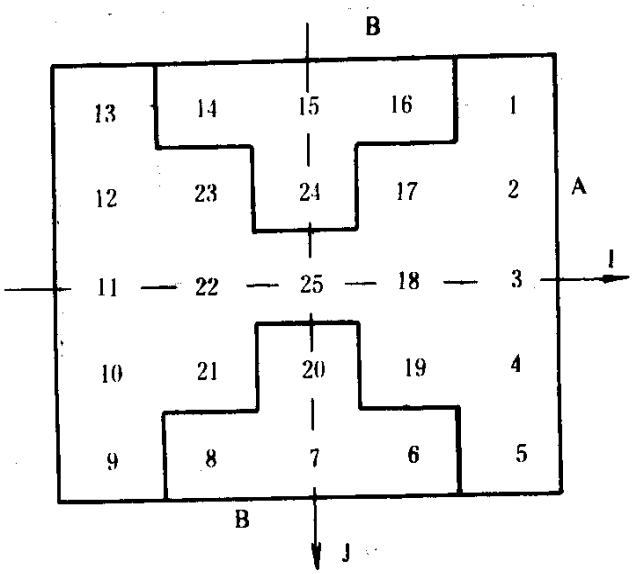


图 9—9　右上顺旋方阵分区图

对照图 9—9 写出通项表达式：

$$\begin{cases} \text{A 区：}N^2-(4I^2+2I+|I-J| & |I|\geqslant|J| \quad (1) \\ \text{B 区：}N^2-(4J^2+2J-|I-J| & |I|<|J| \quad (2) \\ I=m,m-1\cdots\cdots-m+1,-m & (3) \\ J=-m,-m+1\cdots\cdots m-1,m & (4) \end{cases}$$

相应的程序为：

```
40 X=ABS(I)<ABS(J)
50 K=N*N+(4*I*I+2*I+ABS(I-J))*NOT X+(4*J*J+2
   *J-ABS(I-J))*X
```

读者可自己完善程序，程序编号为 No. 9—9。

3. 右下顺旋方阵

右下顺旋方阵的分区图如图 9—10

各区的元素通项表达式为：

$$\begin{cases} \text{A 区：}N^2-(4J+2J+|I+J|) & |I|\leqslant|J| \quad (1) \\ \text{B 区：}N^2-(4I^2-2I-|I+J) & |I|>|J| \quad (2) \end{cases}$$

相应的程序为：

```
40 X=ABS(I)>ABS(J)
```

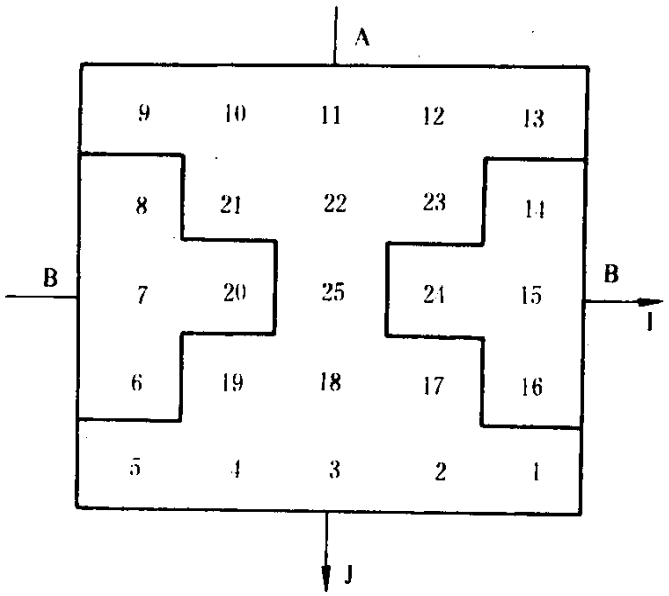


图 9—10

```
50 K=N*N+(4*I*I-2*I-ABS(I+J))*X+(4*J*J+2J+
   ABS(I+J))*NOT X
```

右下顺旋方阵程序编号为 No. 9—10

4. 左下顺旋方阵

左下顺旋方阵的分区图见图 9—11。

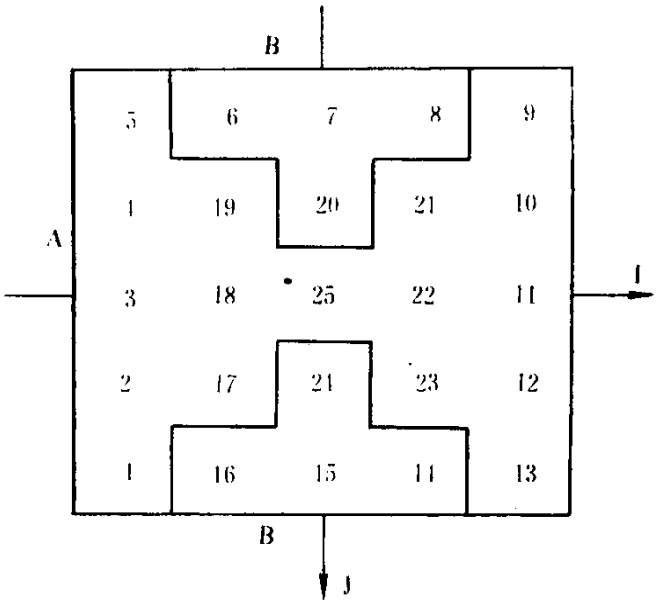


图 9—11　左下顺旋方阵分区图

各区的元素通项表达式为

$$\begin{cases} \text{A 区:}N^2-(4I^2-2I+|I-J|) & |I|\geqslant|J| \quad (1)\\ \text{B 区:}N^2-(4J^2-2J-|I-J| & |I|<|J| \quad (2)\\ I=-m,-m+1\cdots\cdots,m-1,m & (3)\\ J=m,m-1,\cdots\cdots,-m+1,m & (4) \end{cases}$$

对应程序为：

```
40 X=ABS(I)<ABS(J)
50 K=N*N+(4*I*I-2*I+ABS(I-J))*NOT X +(4*J*J-2
   *J-ABS(I-J))* X
```

读者自行合成程序。左下顺旋方阵的程序编号为 No. 9--11。

5. 左上逆旋方阵

以上我们讨论了外向内顺旋方阵的编程技巧。现在讨论外向内逆旋方阵的编程方法。

左上逆旋方阵的区间划分见图 9—12。

各区元素对应的通项表达式为：

$$\begin{cases} \text{A 区:}N^2-(4I^2-2I+|I+J|) & |I|\geqslant|J| \quad (1)\\ \text{B 区:}N^2-(4J^2+2J-|I+J|) & |I|<|J| \quad (2)\\ I=-m,-m+1,\cdots\cdots,m-1,m & (3)\\ J=-m,-m+1,\cdots\cdots,m-1,m & (4) \end{cases}$$

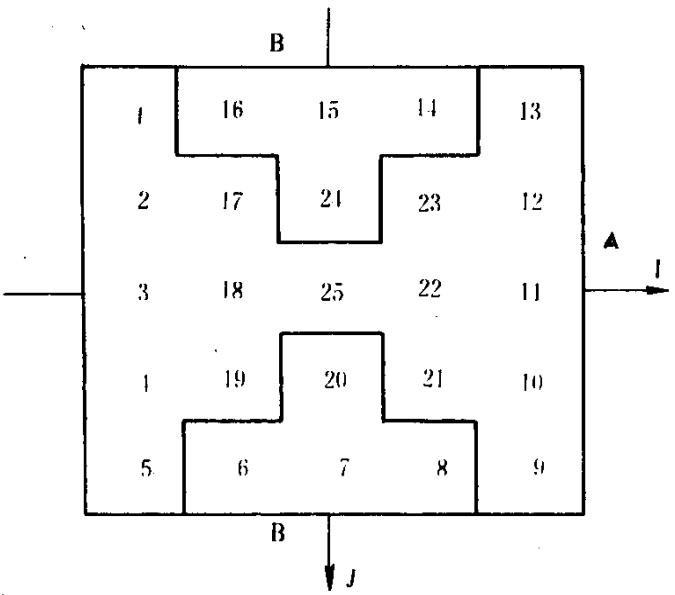


图 9—12　左上逆旋方阵分区图

左上逆旋方阵的程序见 No. 9—12

```
5 REM No. 9—12
10 CLS : INPUT "N=3,5,7,9,";N
20 M=(N-1)/2 : P=3 : IF N>7 THEN P=2
30 FOR I=-M TO M : FOR J=-M TO M
40 X=ABS(I)<ABS(J)
50 K=N * N+(4 * I * I-2 * I+ABS(I+J)) * NOT X+(4 * J * J+2
   * J-ABS(I+J)) * X
60 LOCATE 3 * (N2+I),P * (N/2+J) : PRINT K
70 NEXT : NEXT
80 END
```

可见，顺、逆旋方阵的实现程序结构是相同的，不同的仅仅是40、50两语句行的运算符号稍有变动。

6. 右上逆旋方阵

右上逆旋方阵的区间划分如图 9—13。

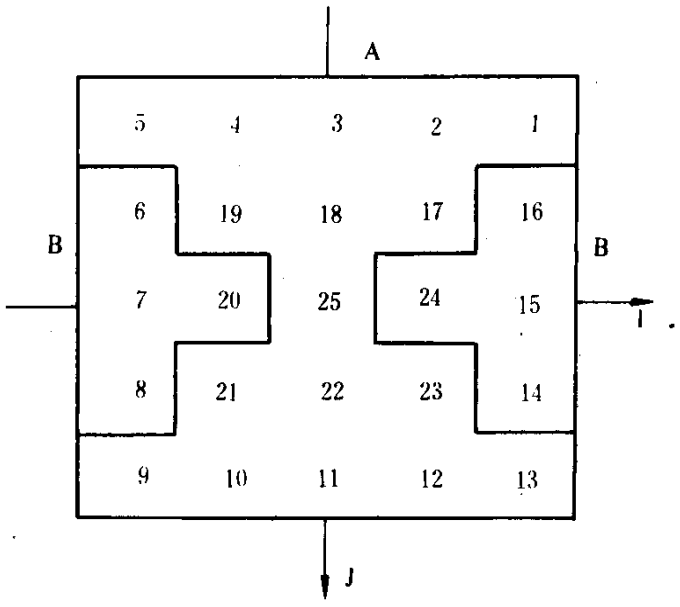


图 9—13　右上逆旋方阵分区图

A、B 区间各元素的通项表达式如下：

$$\begin{cases} \text{A 区:}N^2-(4J^2-2J+|I-J| & |I|\leqslant|J| & (1)\\ \text{B 区:}N^2-(4I^2-2I-|I-J| & |I|>|J| & (2)\\ I=m,m-1,\cdots\cdots,-m+1,-m & & (3)\\ J=-m,-m+1,\cdots\cdots,m-1,m & & (4) \end{cases}$$

对应的 40、50 行程序为：

```
40 X=ABS(I)>ABS(J)
50 K=N*N+(4*I*I-2*I-ABS(I-J))*X+(4*J*J-2*J+
   ABS(I-J))*NOT X
```

由读者自己合成程序。程序编号 No. 9—13。

7. 右下逆旋方阵

右下逆旋方阵的区间划分如图 9—14。

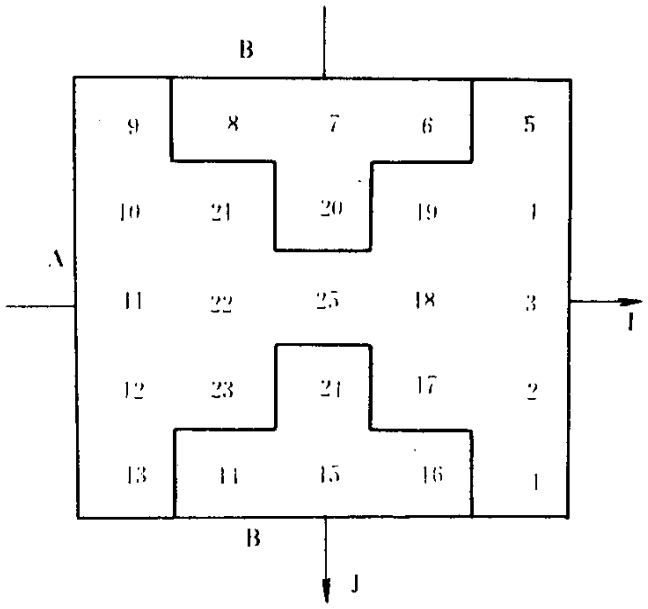


图 9 14 右下逆旋方阵分区图

A、B 区间各元素的通项表达式为：

$$\begin{cases} \text{A 区:}N^2-(4I^2+2I+|I+J| & |I|\geqslant|J| & (1)\\ \text{B 区:}N^2-(4J^2-2J-|I+J|) & |I|<|J| & (2)\\ I=m,m-1,\cdots\cdots,-m-1,-m & & (3)\\ J=m,m-1,\cdots\cdots,-m-1,-m & & (4) \end{cases}$$

对应程序如下：

```
40 X=ABS(I)<ABS(J)
50 K=N*N+(4*I*I+2*I+ABS(I+J))*NOTX+(4*J*J-2*
      J-ABS(I+J))*X
```

由读者自己合成程序，程序编号 No. 9—14。

8. 左下逆旋方阵

左下逆旋方阵的区间划分如图 9—15。

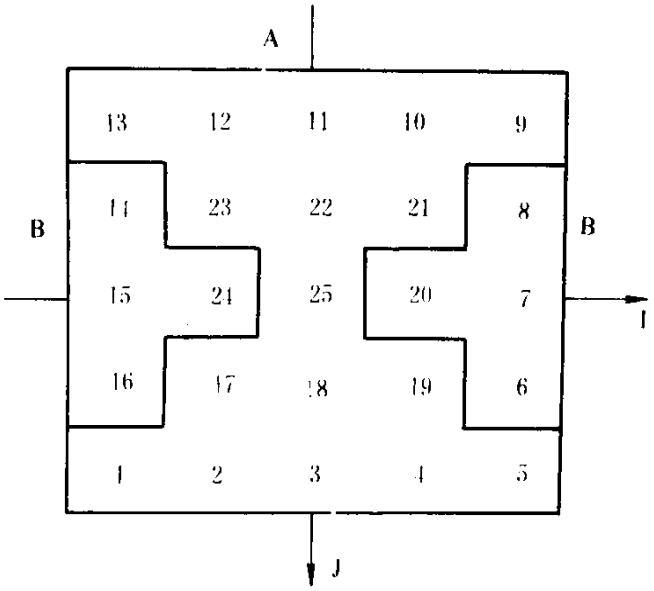


图 9 15 左下逆旋方阵分区图

各区元素的通项表达式为：

$$\begin{cases} \text{A 区：} N^2-(4J^2+2J+|I-J| & |I|\leqslant|J| \\ \text{B 区：} N^2-(4I^2+2I-|I-J|) & |I|>|J| \\ I=-m,-m+1,\cdots\cdots,m-1,m \\ J=m,m-1,\cdots\cdots,-m+1,-m \end{cases}$$

对应程序如下：

```
40 X=ABS(I)>ABS(J)
50 K=N*N+(4*I*I+2*I-ABS(I-J))*X+(4*J*J+2*J+
   ABS(I-J))*NOT X
```

程序编号 No. 9—15。

9. 内向外旋方阵

上面我们讨论了外向内旋方阵的八种形式，及各种形式的区间划分，各区元素的通项表达式和实现的程序。下面再讨论内向外旋方阵的编程方法。我们以右上顺旋方阵为例。

内向外右上顺旋 5 阶方阵的形式如图 9—16。我们仍然采用外向内旋方阵的分析方法。即对方阵分区，写出各区元素的通项表达式，根据通项式编写程序。

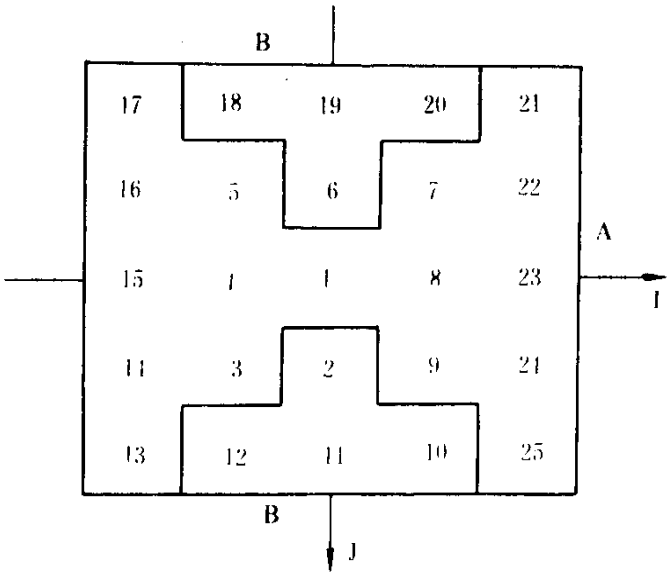


图 9—16　内向外右上顺旋方阵分区图

内向外右上顺旋方阵的区间划分见图 9—16，各区元素的通项表达式如下：

$$\begin{cases} \text{A 区：}1+(4I^2+2I+|I+J| \quad |I|\geqslant|J| & (1) \\ \text{B 区：}1+(4J^2-2J-|I+J| \quad |I|<|J| & (2) \\ I=-m,-m+1,\cdots\cdots,m-1,m & (3) \\ J=-m,-m+1,\cdots\cdots,m-1,m & (4) \end{cases}$$

根据通项表达式编写程序如下：(见 No. 9—16)

```
5 REM No. 9—16
10 CLS : INPUT "INPUT N=3,5,7,9 : N=";N
20 M=(N-1)/2 : P=3 : IF N>7 THEN P=2
30 FOR I=-M TO M : FOR J=-M TO M
40 X=ABS(I)<ABS(J)
50 K=1-(4*I*I+2*I+ABS(I+J))*NOT X-(4*J*J-2*J-
   ABS(I+J))*X
60 LOCATE 3*(N/2+I),P*(N/2+J) : PRINT K
70 NEXT : NEXT : END
```

根据外向内旋的方阵通项式，不难写出内向外旋的后七种形式的通项式，不再赘述。

10. 夹心螺旋方阵

所谓夹心螺旋方阵，是指方阵的外圈是奇数螺旋，第二圈是偶数螺旋，第三圈又是奇数螺旋……。或者反之，外圈是偶数螺旋，第二圈是奇数螺旋……。

外向内左上顺旋5阶夹心方阵的形式见图9—17。

1	3	5	7	9
31	2	4	6	11
29	16	33	8	13
27	14	12	10	15
25	23	21	19	17

图9—17　五阶夹心方阵

我们有了以上各种方阵的编程基础，编写夹心螺旋方阵的程序应是轻车熟路。下面仅给出图9—17的程序清单，见程序No. 9—17。程序结构留给读者分析。

```
5 REM No. 9—17
10 CLS : INPUT "N=3,5,7,9 INPUT N=";N
20 DIM A(N,N),B(2),C(1)
30 B(2)=1 : C(0)=1 : C(1)=2 : T=1 : H=N : P=3
40 IF N>7 THEN P=2
```

```
50 FOR I=1 TO 2
60 FOR J=1 TO H
70 B(I)=B(I)+T : V=ABS(I=1)+1
80 X=0 : IF B(V) MOD 2=0 THEN X=1
90 A(B(1),B(2))=C(X) : C(X)=C(X)+2
100 NEXT
110 IF I=1 THEN H=H-1
120 IF H=0 THEN 150
130 NEXT
140 T=-T : GOTO 50
150 FOR I= TO N : FOR J=1 TO N
160 LOCATE 3*I-2,P*J-P : PRINT A(I,J)
170 NEXT : NEXT : END
```

可以看出，No. 9—17 是程序 No. 9—6 的改进型程序。为压缩篇幅，对于夹心螺旋方阵仅给出以上程序，至于其它 15 种方阵的程序编制，留给读者练习。

9.3 数字魔方

所谓数字魔方，是指具有某些特殊特性的数字方阵。如对角线等积方阵、边、线等和方阵等。下面讨论两种数字魔方的实现程序。

9.3.1 对角线等积方阵

例 9.1 要求设计一个数字方阵。使其具有如下两点特性：

①任意一个正方形，对角线顶点元素的乘积相等。

②任意一个长方形，对角线顶点元素的乘积相等。

根据问题要求，我们使用一个两维数组编程如下（见 No. —18）。

```
5 REM No. 9—18
10 CLS : INPUT : "N=2～9,INPUT N=";N
```

```
20 DIM A(N,N)
30 FOR I=1 TO N
40 FOR J=1 TO N
50 A(I,J)=I*J
60 PRINT A(I,J)"    "
70 NEXT:PRINT :PRINT
80 NEXT:END
```

RUN↙键入5,则打印一个5×5的数字方阵(见图9—18)。

1	2	3	4	5
2	4	6	8	10
3	6	9	12	15
4	8	12	16	20
5	10	15	20	25

图9—18　对角线等积方阵

检查结果是否满足例题要求。

1. 首先检查最大的两条对角线有

$$1\times25=5\times5$$

2. 从方阵的右下角截取一个3×3的方阵如图9—19。有

$$9\times25=15\times15$$

3. 从方阵的中央截取一个3×3的方阵如图9—20。

9	12	15
12	16	20
15	20	25

图9—19

4	6	8
6	9	12
8	12	16

图9—20

有：　　$4\times16=8\times8$

4. 从方阵的右上角截取一个 3×4 的长方形方阵。如图 9—21。

2	3	4	5
4	6	8	10
6	9	12	15

图 9—21

有：　　2×15＝6×5

5. 从方阵的左下角截取 2×3 的方阵如图 9—22。

4	8	12
5	10	15

图 9—22

有：　　4×15＝5×12

可见，程序 No. 9—18 满足了例题 9—11 的要求。

程序 No. 9—18 虽能满足问题要求，但要使用两维数组，占用内存较多。我们也可以使用简单变量，通过巧妙地编程达到同一目的。见程序 No. 9—19。

```
5 REM No. 9—19
10 CLS : INPUT "N=1～9,INPUT N=";N
20 FOR I=1 TO N
30 FOR J=1 TO N * I STEP I
40 PRINT J"    "; : NEXT
50 PRINT : PRINT : NEXT
60 END
```

RUN↙键入"6"，则打印出一个 6×6 的方阵，见图 9—23。

1	2	3	4	5	6
2	4	6	8	10	12
3	6	9	12	15	18
4	8	12	16	20	24
5	10	15	20	25	30
6	12	18	24	30	36

图 9—23　对角线等积方阵

两个程序，结果一致，但程序 No. 9—19 不用开辟数组，程序简洁且节省内存，是一个较理想的程序。

9.3.2　边线等和方阵

在中国古代“算经”中有一道“九宫”的数学题。题解说：

九宫之义，法似灵龟。

六八为肩，二四为足。

左七右三，戴九履一。

五居中央。

意思是说，用 1～9 九个自然数，组成一个三阶方阵，使方阵中每条边的数字之各、对角线的数字之各和均等于 15。见图 9—24。

2	9	4
7	5	3
6	1	8

图 9—24　九宫

题解说明了每个位置放置的数字。

实质上“九宫”就是一个边线等和的 3×3 方阵。能否将其题意扩展，填出 5×5 或 7×7 的边线等和方阵呢？答案是肯定的。

1. 奇数魔方。

编程的设计思路是：根据 H·Coxeter 提出的，当 N 为奇数时生成魔方的简单规则为：

(1)开始时先在第一行中央填写数字“1”；

(2)然后向左上方移动，按自然数顺序将数字填入空格；

(3)当左上方位置离开了方阵时，若从左边离开，便认为是最右一列；若从上方离开，便认为是最下一行；

(4)若左上方已填有数字,便改填正下方空格;

(5)重复(2)～(4),直到填满方阵为止。

根据这些规则,编程如下:见 No. 9—20。

```
5 REM No. 9—20
10 CLS : INPUT "INPUT AN ODD NUMBER";N : PRINT "N="N
   : PRINT
20 IF N MOD 2 =0 THEN PRINT "ERROR;" : PAUSE 100 : GOTO
   10
30 N1=N-1 : DIM A(N1,N1)
40 I=0 : J=N1/2 : A(I,J)=1
50 FOR W=2 TO N*N
60 IF I<>0 THEN 80
70 L=N1 : GOTO 90
80 L=I-1
90 IF J<>0 THEN 110
100 M=N1 : GOTO 120
110 M=J-1
120 IF A(L,M)=0 THEN 140
130 I=I+1-(I+1)/N : GOTO 150
140 I=L : J=M
150 A(I,J)=W
160 NEXT
170 FOR I=0 TO N1 : FOR J=0 TO N1 : PRINT A(I,J);
180 P=P+A(I,J)
190 NEXT : PRINT : PRINT : NEXT
200 PRINT "THE SUM IS " P/N
210 END
```

程序说明:

①10～40 行为初始化程序。10 行要求输入奇数,20 行控制输入偶数时的转向。

②50～160 行完成方阵中各数字位置的确定。

③170～200 行打印方阵和边线之和。

运行程序，键入 3"↙则打印如下方阵(见图 9—25)。

对比图 9—24，程序打印的方阵与九宫题解的结果是相同的，(仅上、下行颠倒了位置)。

若键入 5 则打印出 5×5 的方阵(见图 9—26)。下面给出 N＝7 和 N＝9 的边线等和方阵，请读者验证(见图 9—27，图 9—28)。

```
 6   1   8
 7   5   3
 2   9   4
THE SUM IS 15
```

图 9—25

若去掉 180 行的求和语句，使用 No. 9—20，在 V3 版本中可求 60 以内的方阵，对于 BS. 2A 版本则可求出 40 以内的任意奇数方阵。

```
15   8   1  24  17
16  14   7   5  23
22  20  13   6   4
 3  21  19  12  10
 9   2  25  18  11
THE SUM IS 65
```

图 9—26

```
28  19  10   1  48  39  30
29  27  18   9   7  47  38
37  35  26  17   8   6  46
45  36  34  25  16  14   5
 4  44  42  33  24  15  13
12   3  43  41  32  23  21
20  11   2  49  40  31  22
THE SUM IS 175
```

图 9—27

45	34	23	12	1	80	69	58	47
46	44	33	22	11	9	79	68	57
56	54	43	32	21	10	8	78	67
66	55	53	42	31	20	18	7	77
76	65	63	52	41	30	19	17	6
5	75	64	62	51	40	29	27	16
15	4	74	72	61	50	39	28	26
25	14	3	73	71	60	49	38	36
35	24	13	2	81	70	59	48	37

THE SUM IS 369

图 9—28

程序 No. 9—20 容易理解，但程序稍嫌繁琐，若引入符号函数，可使程序得到极大的简化（见 No. 9—21）。

```
5 REM No. 9—21
10 CLS : INPUT "INPUT AN ODD";N : PRINT
20 X=(N+1)/2 : Y=N
30 FOR I=1 TO N * N : A(X,Y)=I
40 M=X * SGN(N-X)+1 : P=Y * SGN(N-Y)+1
50 A=SGN(A(M,P)) : X=M+A * (X-M) : Y=P+A * (Y--1-
   P) : NEXT
60 FOR I=N TO 1 STEP-1 : FOR J=N TO 1 STEP--1 : PRINT A
   (J,I); : P=P+A(I,J) : NEXT
70 PRINT : PRINT : NEXT : PRINT : PRINT : PRINT "THE SUM
   IS " P/N
```

程序中 10～20 行为人机对话和初始化程序；30～50 行为确定方阵中各元素的座标程序，60～70 行为输出程序。

运行程序可打印出如图 9—24～图 9—26 同样的数字方阵。

程序 No. 9—21 简洁清新，独树一帜，具有很好的启迪性。

2. 奇偶数魔方

以上仅讨论了奇数魔方的构造程序，读者可以试排一下，偶数魔方的构成要比奇数魔方复杂得多。关于构成偶数魔方的原理和过程，由于篇幅较大故略过，仅给出实现的程序。为便于使用，我们

把奇、偶魔方构成程序合二为一，合并成一个边线等和数字方阵构成的通用程序(见 No. 9－22)。

```
5 REM No. 9—22
10 CLS
20 INPUT "N>2";N
30 PRINT :I=1 :P=1:G=N AND 3
40 L=N/(1-(G=2)):J=(L+1)/2:IF G=0 THEN J=1
50 DIM S(L,L):FOR T=1 TO L*L:R=T/L
60 S(I,J)=T:IF G>0 THEN 100
70 IF T MOD I=0 AND T/L=R THEN I=I+1:J=(1+L)*SGN(I
   AND 2)
80 P=1-P :IF P=0 THEN I=L-I+1
90 J=J+1-(R+1 AND 2):GOTO 130
100 IF T MOD L=0 AND T/L=R THEN I=I+1:GOTO130
110 I=I-1-(I=1)*L
120 J=J+1+(J=L)*L
130 NEXT:T=T-1
160 IF N>L THEN 230
170 FOR I=1 TO N:FOR J=1 TO L
180 PRINT S(I,J);:NEXT:PRINT:PRINT:NEXT
190 PRINT:PRINT:"A="(N*N+1)*N/2
200 PRINT:INPUT:"AGAIN PRESS <Y>";Y$
210 IF Y$="Y" THEN RUN 20
220 END
230 R=L/2+1:FOR S=1 TO N:I=S+(S>L)*L
240 FOR P=1 TO N:J=P+(P>L)*L:G=-(J>1 OR I<>R)
250 G=3*T*(ABS(S>L)<>G*ABS(J<R)+(I=R)*(J=R))
260 IF P>L THEN G=T+ABS((S>L)=(1+J<R))*T
270 PRINT S(I,J)+ABS(G);
280 NEXT:PRINT:PRINT:NEXT:GOTO 190
```

程序说明：

①10～40 为人机对话和初始化程序。

②50 行中定义数组 S(L,L)为输出魔方作准备。

③70～90 行作用是完成 4 的倍数阶魔方的构造。

④100～130 行完成其它的情况的魔方设计。

⑤160～190 行是奇数魔方的输出打印程序。

⑥200～220 行为结束处理程序。

⑦230～280 行的程序完成偶数魔方的换算合成及输出。

运行程序，输入奇数，将打印如图 9—24～图 9—26 完全相同的奇数魔方。若输入 4 或 6 的偶数，将输出如图 9—27、图 9—28 的偶数阶魔方。

```
 1   14   15   4
 8   11   10   5
12   7    6    9
13   2    3    16
A=34
```

图 9—27　4 阶魔方

```
35  1   6   26  19  24
3   32  7   21  23  25
31  9   2   22  27  20
8   28  33  17  10  15
30  5   34  12  14  16
4   36  29  13  18  11
A=111
```

图 9—28　6 阶魔方

9.4　八皇后问题

八皇后问题是伟大的数学家高斯提出来的。它可以表述为：在

一张国际象棋的棋盘上摆八个皇后，使其互不威胁。即没有两个皇后处于同一行、同一列、同一对角线上。求共有多少种摆法。

皇后问题的示意图见图 9—29。

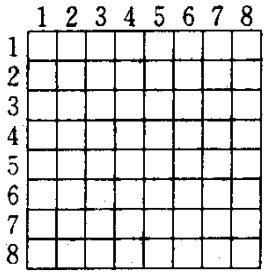


图 9—29　八皇后问题示意图

我们把棋盘的行与列分别标上号码，使每个方格都可以用唯一的座标(X,Y)表示。所谓对角线就是指由行号码与列号码之差相等的方格组成，或由行号码与列号码之和相等的方格组成。可以算出图 9—29 共有 30 条对角线。每一个方格对应一行、一列、两条对角线。

9.4.1　八皇后问题的一般解法

求解八皇后问题的一般思路是：首先从第一行开始，尝试在第一列摆上第一个皇后，摆好这个皇后以后，就将其相应格的行、对角线设禁摆标志，并下移一行，这样就保证了所摆的皇后绝对不同行。然后，在第二行逐列尝试，寻找安全的方格，只要保证当前摆下的皇后与已摆的那些皇后不在同一列和同一对角线上即可。判断某方格是否可放皇后，是通过检查相应的列和对角线和标志实现的。如此，一直继续到所有的皇后都摆完为止。

讨论程序的编写之前，还要介绍一个重要的概念——回溯。回溯概念的具体描述如下：在上述思路的求解过程中，可能会出现在某行找不到一个安全方格的情况。这时就要返回到上一行，取消已摆的皇后，改放在下一列继续尝试。这一返回到上一行的过程就是“回溯”。另外，当我们求出一组解后，还要求其它所有组解，这就需要将所有可能的摆法都判断一下。这也需要程序有回溯功能。即

在求出一组解后，再接着在最后一行的下一列继续尝试下去。整个求解的终结条件是：列号加到 8，且行号减到 0。

根据上述讨论，确定编程思路如下：

1. 定义列标志数组 H(8)、列数组 B(8)和对角线标志数组 F(30)；X＝1、Y＝1. 确定第一方格等初始化。

2. 若方格的列和对角线标志值至少有一为禁摆标志(非零)，则转 6。

3. B(X)＝Y。

4. 若 X 值小于 8 则转去 12。

5. 输出一组解[(1,B(1)),(2,B(2)),……(N,B(N))]，解数变量 T 累加 1。

6. 若 Y 值小于 8 则转去 11。

7. X＝X—1。

8. 若 X≠0，则转 10。

9. 若 X＝0 则结束。输出解的累计数。

10. Y＝B(X)，将(X,Y)相应的列和对角线标志清零，释放为以下可选的方格(自由方格)，转去 6。

11. Y＝Y＋1，转去 2。

12. 置方格的列和对角线标志为非零，表示禁摆皇后。

13. X＝X＋1，Y＝1，转去 2。

其中第 7、8、10 三步为回溯功能块。

根据以上编程思路，编写程序如下：(见 No. 9—23)

```
5 REM No. 9—23
10 CLS : DIMB(8),H(8),F(30) : PRINT "Please wait!"
20 D=8 : U=14 : X=1 : Y=1
30 I=D+X-Y : J=U+X+Y
40 IF H(Y)=1 OR F(I)=1 OR F(J)=1 THEN 80
50 B(X)=Y
```

```
60 IF X<8 THEN H(Y)=1 : F(I)=1 : F(J)=1 : X=X+1 : Y=1 :
   GOTO 30
70 T=T+1 : PRINT "No." T ",[";: FOR I=1 TO 8 : PRINT "("I
   "," B(I) ")";: NEXT : PRINT "]" : PRINT
80 IF Y<8 THEN Y=Y+1 : GOTO 30
90 X=X-1 : IF X<0 THEN PRINT "TOTAL=";T : END
100 Y=B(X) : H(Y)=0 : F(D+X-Y)=0 : F(U+X+Y)=0 : GO-
    TO 80
RUN↙打印
Please wait!
No. 1. [(1,1)(2,5)(3,8)(4,6)(5,3)(6,7)(7,2)(8,4)]
No. 2. [(1,1)(2,6)(3,8)(4,3)(5,7)(6,4)(7,2)(8,5)]
No. 3. [(1,1)(2,7)(3,4)(4,6)(5,8)(6,2)(7,5)(8,3)]
No. 4. [(1,1)(2,7)(3,5)(4,8)(5,2)(6,4)(7,6)(8,3)]
No. 5. [(1,2)(2,4)(3,6)(4,8)(5,3)(6,1)(7,7)(8,5)]
No. 6. [(1,2)(2,5)(3,7)(4,1)(5,3)(6,8)(7,6)(8,4)]
No. 7. [(1,2)(2,5)(3,7)(4,4)(5,1)(6,8)(7,6)(8,3)]
No. 8. [(1,2)(2,6)(3,1)(4,7)(5,4)(6,8)(7,3)(8,5)]
No. 9. [(1,2)(2,6)(3,8)(4,3)(5,1)(6,4)(7,7)(8,5)]
No. 10. [(1,2)(2,7)(3,3)(4,6)(5,8)(6,5)(7,1)(8,4)]
No. 11. [(1,2)(2,7)(3,5)(4,8)(5,1)(6,4)(7,6)(8,3)]
No. 12. [(1,2)(2,8)(3,6)(4,1)(5,3)(6,5)(7,7)(8,4)]
No. 13. [(1,3)(2,1)(3,7)(4,5)(5,8)(6,2)(7,4)(8,6)]
No. 14. [(1,3)(2,5)(3,2)(4,8)(5,1)(6,7)(7,4)(8,6)]
No. 15. [(1,3)(2,5)(3,2)(4,8)(5,6)(6,4)(7,7)(8,1)]
No. 16. [(1,3)(2,5)(3,7)(4,1)(5,4)(6,2)(7,8)(8,6)]
No. 17. [(1,3)(2,5)(3,8)(4,4)(5,1)(6,7)(7,2)(8,6)]
No. 18. [(1,3)(2,6)(3,,2)(4,5)(5,8)(6,1)(7,7)(8,4)]
No. 19. [(1,3)(2,6)(3,2)(4,7)(5,1)(6,4)(7,8)(8,5)]
N0. 20. [(1,3)(2,6)(3,2)(4,7)(5,5)(6,1)(7,8)(8,4)]
  ⋮
```

TOTAL＝92

从以上部分运行结果可以明显看出，计算机的求解过程就是从第一行第一列开始逐个尝试，确定皇后摆放的位置的。

程序说明：

①10～20 行为初始化程序，它分别定义了列数组 B(8)、列标志数组 H(8)和对角线标志数组 F(30)，并确定第一个皇后的尝试摆放位置为(1,1)。其中 D、U 两个常数是对角线换算变量。X、Y、对角线的排列顺序见图 9—30。

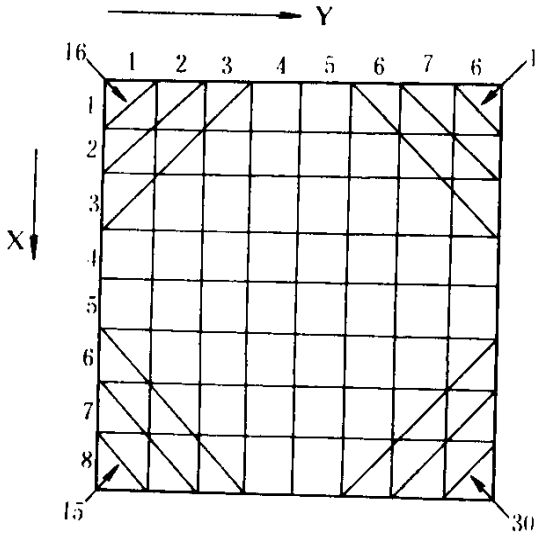


图 9—30　程序 No. 9—23X、Y 对角线排列顺序

从图 9—30 可见，对角线的排列从棋盘的右上角开始，以左上→右下斜线为第一条，一直排到左下角为第 15 条，然后再从左上角开始，排到右下角为第 30 条。程序 10～20 行对应编程思路中的第一步。

②30～80 行为尝试摆放皇后的主程序。其中 40 行对应第二步；50 行对应第三步；60 行对应第四、十二、十三步；70 行对应第五步；80 行对应第十一步。该段程序的执行过程如下：根据 X、Y 的初值由 50 行决定摆放皇后的列位置，随后由 60 行对该位置设

置禁摆标志，如(1,1)摆皇后，则设第一列禁摆 H(1)＝1，第八条、第十六条对角线禁摆 F(8)＝1、F(16)＝1。然后 X＝X＋1，即下移一行，再从第一列开始尝试摆第二个皇后(返回 30 行)。通过 30 行计算对角线编号后，由 40 行检查该方格(2,1)是否设禁摆标志，若有，(即 H(Y)＝H(1)＝1)则转 80 行换一列(Y＝Y＋1)后，返回 30 行继续尝试，直到找出安全的方格。

③90～100 行为回溯程序，它分别对应第七、八、十步。当在某一行的各列都没有安全方格时，则由 90 行令 X＝X－1，即返回上一行，否定上一行已摆皇后的位置，同时由 100 行解除禁摆标志，然后在上一行重新尝试摆放皇后的位置。若在上一行也找不到安全位置则继续到上二行。它可一直回溯到第一行(X＝1)，若仍找不到安全方格则结束运行。所以说当列号增到 8、行号减到 0 时为结束标志，即 90 行的第二语句对应第九步。

9.4.2 八皇后题解的动态模拟

通过 9.4.1 的讨论，我们找出了八皇后问题的 92 组解。可以逐一验证，这 92 组解都是正确的。但是这一验证工作虽然简单却很麻烦，因为起码要画 92 张棋盘，并对照运算结果分别填入 736 个皇后，这无疑是一项繁琐的工作，能否让计算机画出棋盘，自动摆放好每张棋盘的 8 个皇后供我们验证呢？程序 No. 9－24 可满足这一要求。

首先按图 9－31 进行键控绘图。然后输入下列程序。

```
5 REM No. 9—24
10 DIM C(8),T(8),S(15),P(15) : D=1
20 SPRITE ON : VIEW
30 FOR I=0 TO 7 : DEF SPRITE I,(I/2,1,0,0,0)=CHR $ (28+I *
   4－J)+CHR $ (29+I * 4－J)+CHR $ (30+I * 4－J)+CHR $ (31
   +I * 4－J)
40 IF I=6 THEN J=20
50 NEXT
```

```
60 K=1
70 IF T(K)=1 OR S(8+D-K)=1 OR P(D+K-1)=1 THEN 110
80 C(D)=K : T(K)=1 : S(8+D-K)=1 : P(D+K-1)=1 : GO-
   SUB 140
90 D=D+1 : IF D>8 THEN TA=TA+1 : LOCATE 0,22 : PRINT
   "Result" TA : A$ : INKEY$(0) : GOTO 120
100 GOTO 60
110 K=K+1 : IF K<=8 THEN 70
120 D=D-1 : IF D<1 THEN SPRITE OFF : CLS : PRINT "TO-
    TAL=" TA : END
130 K=C(D) : T(K)=0 : S(8+D-K)=0 : P(D+K-1)=0 : GO-
    TO 110
140 X=40+(K-1)*24 : Y=60+(D-1)*16
150 SPRITE D-1,X,Y : RETURN
```

运行程序，显示图 9－32 的画面。从画面上可以清楚地看到程序的回溯过程，有关的皇后不停地变换着自己所在位置，直到各自都找到安全的方格后，屏幕下方打印结果 1 的提示(Result 1)，各皇后静止不动，屏幕左下角出现闪烁光标。现在，我们可以对照程序 No. 9－23 验证各皇后的摆放位置是否符合规则。验证完毕，按任一键，各皇后又运动演示各自的尝试摆放过程，一直到再次找到各自的安全方格静止，由我们进行第二个结果的验证。只要你有耐心可一直验证完 92 组答案。

程序 No. 9－24 不但为我们验证结果提供了方便，而且还逼真地演示了程序运行过程，形象直观地表现了每个皇后的每次尝试和每一次的回溯，对我们加深程序理解提供了极大的帮助。

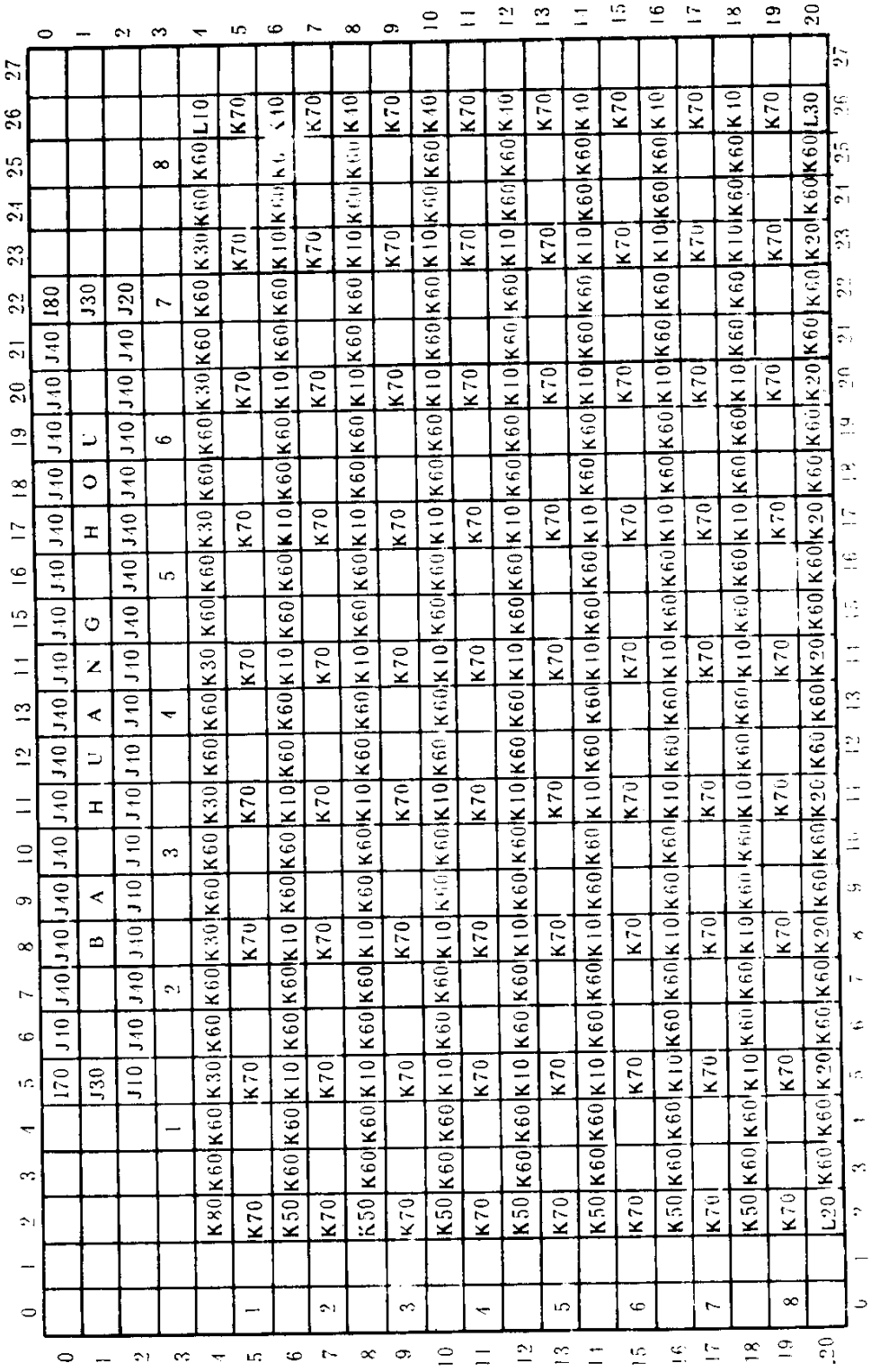


图 9—31　八皇后背景绘图规则

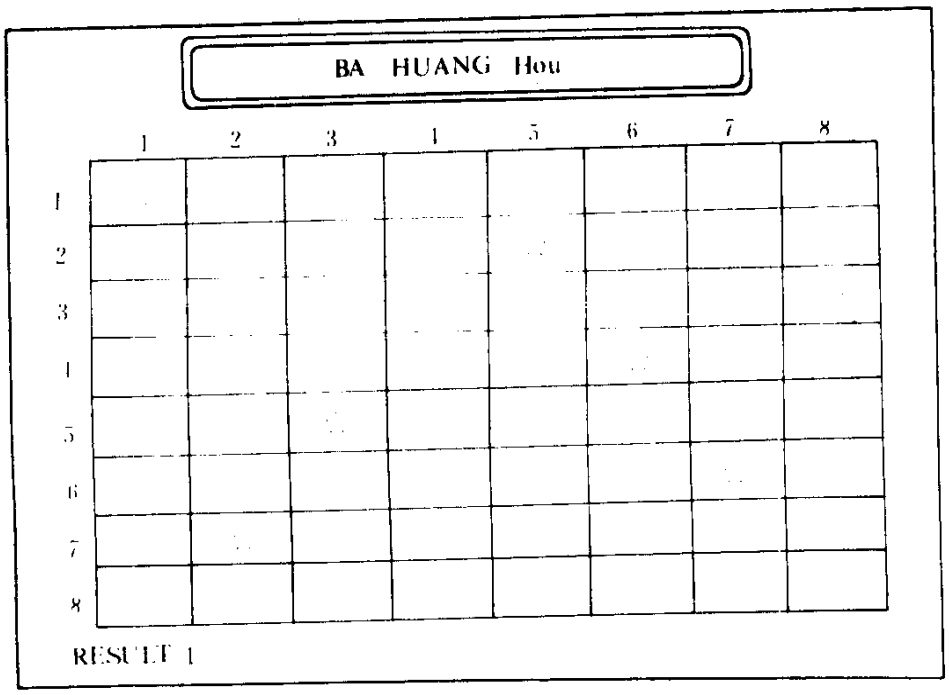


图 9—32　八皇后题解动态演示画面

程序说明：

①10～20 行为初始化程序。10 行定义了 4 个数组，C(8)相当于 No. 9－23 中的 B(8)，T(8)相当于 H(8)，S(15)，P(15)相当于 F(30)。

②30～50 行定义了丽莎小姐的七种动作分别代表 8 个皇后，其中通过变量 J 控制使第 8 个皇后与第一个皇后是同一姿式。

③60～130 行与 No. 9－23 中的 30～100 行的程序结构，作用相同，不多赘述。其中 90 行引入读键盘语句的目的是为用户提供控制程序运行的手段。

④140～150 行的作用是令 8 个皇后在各自的位置上显示。

9.4.3　八皇后问题的推广

以上我们讨论了在 8×8 的棋盘上摆放八个皇后的求解程序。思想活跃的朋友可能会提出更进一步的问题，如：若在 6×6 格的棋盘上摆六个皇后将有多少解；若在 9×9 格的棋盘上摆九个皇后将有多少解；更有甚者还会提出在 20×20 格摆 20 个或对任意自

然数 N×N 的棋盘上摆 N 个皇后有多少解；还可能提出在不同行、列的棋盘上摆与行、列数不同的皇后将有多少解等等问题。现在我们分两步讨论以上问题。

①N×N 格摆 N 个皇后

对任意自然数 N，求解 N×N 格摆 N 个皇后问题，我们可对程序 No. 9－23 进行改造，改造后的程序见 No. 9－25。

```
5 REM No. 9—25
10 CLS : INPUT "N="; N : DIM B(N),H(N),F(4 * N--2)
20 D=N : U=2 * N－2 : X=1 : Y=1
30 I=D+X－Y : J=U+X+Y
40 IF H(Y)=1 OR F(I)=1 OR F(J)=1 THEN 80
50 B(X)=Y
60 IF X<N THEN H(Y)=1 : F(I)=1 : F(J)=1 : X=X+1 : Y=1 :
   GOTO 30
70 T=T+1 : PRINT "No" T"[";: FOR I=1 TO N : PRINT "(" I
   "," B(I) ")";: NEXT : PRINT "]" : PRINT
80 IF Y<N THEN Y=Y+1 : GOTO 30
90 X=X－1 : IF X=0 THEN PRINT "TOTAL=" T : END
100 Y=B(X) : H(Y)=0 : F(D+X－Y)=0 : F(U+X+Y)=0 : GO-
    TO 80
RUN        ↙键入 1 则打印
N=1
No,1,[(1,1)]
TOTAL=1
 若键入 2，则打印
N=2
TOTAL=0
 若键入 3 则打印
N=3
TOTAL=0
```

运行结果表明 N＝2、N＝3 时没有解，事实也的确如此。

若键入 4，则打印

N＝4

No. 1.[(1,2)(2,4)(3,1)(4,3)]

No. 2.[(1,3)(2,1)(3,4)(4,2)]

TOTAL＝2

若键入 5 则打印 10 个结果。

若键入 6 则打印 4 个结果。

若键入 9 则打印 352 个结果。为便于对照，兹录下 N＝9 时的前 10 个答案供参考。

N＝9

No. 1.[(1,1)(2,3)(3,6)(4,8)(5,2)(6,4)(7,9)(8,7)(9,5)]

No. 2.[(1,1)(2,3)(3,7)(4,2)(5,8)(6,5)(7,9)(8,4)(9,6)]

No. 3.[(1,1)(2,3)(3,8)(4,6)(5,9)(6,2)(7,5)(8,7)(9,4)]

No. 4.[(1,1)(2,4)(3,2)(4,8)(5,6)(6,9)(7,3)(8,5)(9,7)]

No. 5.[(1,1)(2,4)(3,6)(4,3)(5,9)(6,2)(7,8)(8,5)(9,7)]

No. 6.[(1,1)(2,4)(3,6)(4,8)(5,2)(6,5)(7,3)(8,9)(9,7)]

No. 7.[(1,1)(2,4)(3,7)(4,3)(5,8)(6,2)(7,5)(8,9)(9,6)]

No. 8.[(1,1)(2,4)(3,7)(4,9)(5,2)(6,5)(7,8)(8,6)(9,3)]

No. 9.[(1,1)(2,4)(3,8)(4,3)(5,9)(6,7)(7,5)(8,2)(9,6)]

No. 10.[(1,1)(2,5)(3,2)(4,6)(5,9)(6,3)(7,8)(8,4)(9,7)]

读者也可以把程序 No. 9－24 改造，进行运算过程的动态模拟。

另外，当 N＞9 时，程序的求解时间将迅速增加。从动态模拟中可清楚地看到回溯次数明显增多。程序 No. 9－25 从理论上可以求 N＝8000 以内的解，但由于受电脑内存的限制，实际的求解能力要小得多。

②M×L 格摆 N 个皇后

求解在 M×L 格的棋盘上摆 N 个皇后的问题，可分两步进

行。第一步：在M行中任选出N行不动；第二步在L列中任选N列与第一步所选的N行作各种不同的组合(即对所选的N列作排列)。最后将满足条件的各点打印出来。程序见No. 9－26。

```
5 REM No. 9—26
10 INPUT "M,L,N,N≤L,N≤M";M,L,N : DIM A(M),B(L) : IF
   N>M OR N>L THEN CLEAR : GOTO 10
20 T=1
30 IF T=0 THEN END
40 A(T)=A(T)+1 : IF A(T)>M THEN A(T)=0 : T=T－1 : GO-
   TO 30
50 I=0
55 IF A(T)<=A(I) THEN 40
60 I=I+1 : IF I<T THEN 55
70 IF T<N THEN T=T+1 : GOTO 40
80 K=1
90 IF K=0 THEN 40
100 B(K)=B(K)+1 : IF B(K)>L THEN B(K)=0 : K=K－1 : GO-
    TO 90
110 I=0
120 IF I=0 THEN 150
130 IF B(I)=B(K) OR ABS(A(I)－A(K))=ABS(B(I)－B(K))
    THEN 100
140 GOTO 160
150 IF B(K)=B(I) THEN 100
160 I=I+1 : IF I<K THEN 120
170 IF K<N THEN K=K+1 : GOTO 100
180 P=P+1 : PRINT "No." P "␣";
190 FOR I=1 TO N : PRINT "(" A(I) "," B(I) ")";:NEXT
200 PRINT : GOTO 100
```

程序说明：

①A(M)为行元素数组，B(L)为列元素数组。要求输入的皇后数要少于或等于行、列数。

②20～70 行的作用是在 M 行中任取 N 行。

③80～170 行的作用是在 L 列中任取 N 列进行排列。其中 130 行保证摆放的各皇后不在同一列和同一对角线，120、150 行保证皇后在边界时不同列。

④180～190 行输出符合条件的摆法。

⑤200 行保证求出所有的解。

运行程序，根据提示依次输入三个数据，便可求出对应的解。

如输入“5,3,2”则打印出 38 个结果。

如输入“5,3,3”则打印如下结果：

```
No. 1    (1,1)(2,3)(4,2)
No. 2    (1,3)(2,1)(4,2)
No. 3    (1,1)(2,3)(5,2)
No. 4    (1,3)(2,1)(5,2)
No. 5    (1,2)(3,1)(4,3)
No. 6    (1,2)(3,3)(4,1)
No. 7    (1,1)(3,2)(5,3)
No. 8    (1,3)(3,2)(5,1)
No. 9    (1,2)(4,1)(5,3)
No. 10   (1,2)(4,3)(5,1)
No. 11   (2,1)(3,3)(5,2)
No. 12   (2,3)(3,1)(5,2)
No. 13   (2,2)(4,1)(5,3)
No. 14   (2,2)(4,3)(5,1)
```

如输入“6,6,6”则打印与程序 No. 9－25 相同的结果如下：

```
No. 1    (1,2)(2,4)(3,6)(4,1)(5,3)(6,5)
No. 2    (1,3)(2,6)(3,2)(4,5)(5,1)(6,4)
No. 3    (1,4)(2,1)(3,5)(4,2)(5,6)(6,3)
```

No. 4　(1,5)(2,3)(3,1)(4,6)(5,4)(6,2)

如输入"6,6,5"则打印出 248 组结果。最后的 5 组结果如下：

No. 244　(2,5)(3,3)(4,1)(5,6)(6,4)

No. 245　(2,5)(3,3)(4,6)(5,4)(6,2)

No. 246　(2,6)(3,2)(4,5)(5,1)(6,4)

No. 247　(2,6)(3,3)(4,5)(5,2)(6,4)

No. 248　(2,6)(3,4)(4,2)(5,5)(6,3)

可以验证，这些结果都是正确的。

由以上运行结果可见，棋盘不变，若皇后减少一个，则摆放方法要极大地增加。如在 6×6 格棋盘中放 6 个皇后仅有 4 种摆法；当摆 5 个皇后时，摆放方法则剧增到 248 种，是摆 6 皇后的 62 倍。有兴趣的读者可试验一下在 8×8 格的棋盘中摆 7 个皇后将有多少种摆法。

另外，程序 No. 9－26 是以排列组合为出发点编写的，故数据间比较次数多，程序的运行速度较慢。

9.5　金叶古塔搬移

传说在印度的佛教圣地贝拿勒斯的圣庙里，安放着一块黄铜板，板上插着三根宝石针，在其中的一根针上从上到下放着由大到小的 64 个金片，这就是所谓的梵塔，也称为金叶古塔。印度佛教的主神梵天要求他的僧侣们把这些金片在三根针上移来移去。并且规定：每次只能移一片，且不管在那一根针上，小片永远在大片的上面。梵天预言：当所有的金片从开始的那根针上（不妨称为起始针座）移到另一根针（目标针座）时，世界的末日就来临了。那么，64 个金片全部移完需要多少时间呢？也就是说，最少要移动多少次才能移完。这在数学界被称之为梵塔问题，也称为金叶古塔搬移问题。中华学习机已有金叶古塔搬移的游戏软件，我们的电脑由于内

存较小，不可能移植这一游戏，但我们可以编程运算，了解金叶古塔搬移的过程。

9.5.1 金叶古塔搬移的一般解法

在许多讲授计算机程序设计的书中，均有金叶古塔搬移的程序介绍，但基本上都是采用递归算法。在F BASIC语言中，由于没有过程调用指令，所以只能使用GOSUB、RETURN语句来模拟实现递归。但这样编程受到计算机堆栈层数的限制，往往不能如愿。下面我们讨论一种金叶古塔搬移的非递归算法。

首先，我们以4个金叶为例，分析讨论目标针座为3时的古塔移动过程(见图9－33)。

对照图9－33我们可写出各金叶的移动过程(见表9－3)。

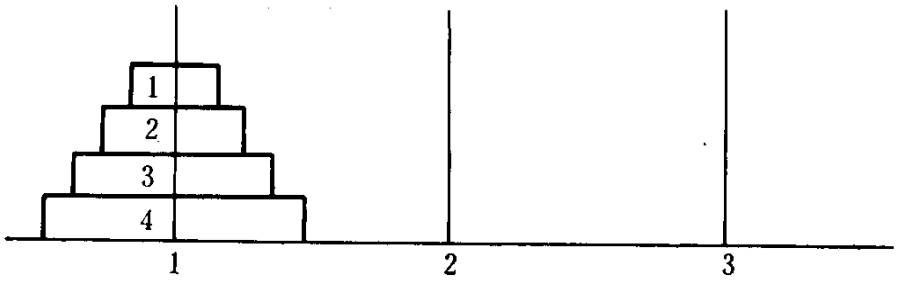


图9—33 四层古塔

表9－3 四层古塔目标针座为3的搬移过程

移动顺序号	金叶编号	移出针座编号	移入针座编号
1	1	1	2
2	2	1	3
3	1	2	3
4	3	1	2
5	1	3	1
6	2	3	2
7	1	1	2
8	4	1	3
9	1	2	3
10	2	2	1
11	1	3	1
12	3	2	3
13	1	1	2
14	2	1	3
15	1	2	3

仔细观察表 9－3 的移动过程，不难发现，各金片的移动是相当有规律的。4 个金片一共移动了 15 次(2^4-1)。其中 1 号金片移动了 8 次；2 号金片移动了 4 次；3 号金片移动了 2 次；4 号金片移动了 1 次。它们的移动规律如表 9－4。

表 9－4　四层古塔 3 号针座的移动规律

金片号	在第几步时移动	通项表达式
1	1、3、5、7、9、11、13、15	$2^0(2K+1)$：$K=0\sim2^3-1$
2	2、6、10、14	$2^1(2K+1)$：$K=0\sim2^2-1$
3	4、12	$2^2(2K+1)$：$K=0\sim2^1-1$
4	8	$2^3(2K+1)$：$K=0\sim2^0-1$

用表 9－4 中的通项表达式，我们可以算出

在第 $2^0(2K+1)$步时，1 号金片在移动；

在第 $2^1(2K+1)$步时，2 号金片在移动；

在第 $2^2(2K+1)$步时，3 号金片在移动；

在第 $2^3(2K+1)$步时，4 号金片在移动。

一般地，N 个金片要移动 2^N-1 步，在第 $2^S(2K+1)$步时，S＋1 号金片在移动($S=0,1,2\cdots\cdots,N-1$)。

其次，分析讨论每个金片移动的规律。

从表 9－3 可见，1 号金片尽管移动了 8 次，但它只是翻来复去地执行一个基本的动作(见图 9－34)：

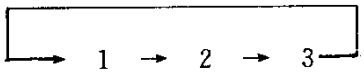


图 9－34

同样 2 号金片则执行另一个基本的动作(见图 9－35)：

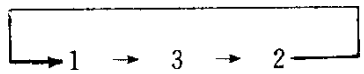


图 9－35

通过对不同个数的金片以及不同的目标针座(2 或 3)进行试验观察，总结出如下规律：

设金片个数为N，目标针座号为M，则有：

①“当N为奇数且M＝2”或“当N为偶数且M＝3”时，奇数号金片按图9－34的顺序移动；偶数号金片按图9－35的顺序移动。

②“当N为奇数且M＝3”或“N为偶数且M＝2”时，奇数号金片按图9－35的顺序移动；偶数号金片按图9－34的顺序移动。

为了区别这两种情形，我们引进一个K标志：在情形①时，令K＝1；在情形②时，令K＝2。

最后，再分析讨论在第几步时几号金片从哪一个针座移到另外一个针座。

可以证明，任意一个自然数N都可以表示为

$$N=2^S(6W+Y)$$

的形式。其中S＝0，1，2……

W＝0，1，2……

Y＝1，3，5。

如：

$10=2^1(6\times0+5)$

$17=2^0(6\times2+5)$

$36=2^2(6\times1+3)$

$100=2^2(6\times4+1)$

……

所以，移动古塔的每一步总可以转成为$2^S(6W+Y)$的形式。

为了区分奇、偶数金片，我们还需引进一个P标志：若S为奇数则令P＝1；若S为偶数则令P＝2。

这样，由S的值可以确定是S＋1号金片在移动，由K、P、Y的值可以确定S＋1号金片的移动方向。S＋1号金片的移动方向的确定规则如下：

当K·P≠2时，
- Y＝1，作1→3的移动
- Y＝3，作3→2的移动
- Y＝5，作2→1的移动

$$\text{当 K·P=2 时,}\begin{cases}Y=1\text{,作 1→2 的移动}\\Y=3\text{,作 2→3 的移动}\\Y=5\text{,作 3→1 的移动}\end{cases}$$

为了减少程序设计中过多的分支跳跃，增强程序的可读性及加快执行速度，可对上述移动规则实行进一步的简化：

$$\text{当 K·P≠2 时令}\begin{cases}A=Y+3\cdot(Y=5)\\B=(7-Y)/2\end{cases}$$

则 S+1 号金片从 A 移动 B；

当 K·P=2 时，令$A=(Y+1)/2$

$$B=A+1+3\cdot(Y=5)$$

则 S+1 号金片从 A 移到 B。

至此，我们已对梵塔的移动过程进行了彻底的分析讨论，对于任意个金片和指定的目标针座(2 或 3)，我们都可以准确地指出在第几步时是哪一个金片在作怎样的移动。在此基础上，确定我们的编程思路如下：

1. 确定金片的个数 N 和目标针座 M。

2. 根据 N 的奇偶和 M 的值确定 K 标志。

若“N 为奇数且 M=2”或“N 为偶数且 M=3”则令 K=1；

若“N 为奇数且 M=3”或“N 为偶数且 M=2”则令 K=2。

3. 对梵塔移动的某一步 I(I=1,2,3,……,2^N-1)作以下三项工作：

(1)将 I 转化成 $2^S(6W+Y)$的形式，即令

$$I=2^S(6W+Y)$$

(2)由 S 的奇、偶确定 P 标志：

若 S 为奇数，令 P=1；

若 S 为偶数，令 P=2。

(3)当 K·P≠2 时，令$A=Y+3\cdot(Y=5)$

$$B=(7-Y)/2$$

则 S+1 号金片从 A 移到 B；

例如：设 N＝5，目标针座 M＝2，判断第 20 步(I＝20)哪一个金片做怎样的移动。

解：由于 N 为奇数且 M＝2 则 K＝1；将 20 转化为 $20=2^2(6\times 0+5)$，由 S＝2 推知第 3 号金片在移动，且 P＝2；由 K·P＝2 且 Y＝5 可知：A＝3，B＝1。因此，知道第 20 步应为第 3 号金片从 3 号针座移到 1 号针座。

4. 现在，我们有了明确的数字表达式和逻辑表达式，只要设置一重循环，令 I 从 1 到 2^N-1 变化，就可求出 N 片金叶的全部移动过程(程序见 No. 9—27)。

```
5 REM No. 9—27
10 CLS : INPUT "Discs=";N
20 A$ ="Disc" : B$ ="from" : C$ ="to"
30 INPUT "Seats(2 or 3)";M : PRINT
40 IF M<>2 AND M<>3 THEN 30
50 IF N MOD 2<>0 AND M=2 OR N MOD 2=0 AND M=3 THEN
   K=1
60 IF N MOD 2<>0 AND M=3 OR N MOD 2=0 AND M=2 THEN
   K=2
70 J=1 : FOR I=1 TO N : J=J * 2 : NEXT : J=J-1
80 FOR I=1 TO J : W=I : S=0 : P=2
90 IF W MOD 2=0 THEN W=W/2 : S=S+1 : GOTO 90
100 Y=W MOD 6 : PRINT "("I")";
110 IF S MOD 2<>0 THEN P=1
120 IF K * P<>2 THEN A=Y+3 * (Y=5) : B=(7-Y)/2
130 IF K * P=2 THEN A=(Y+1)/2 : B=A+1+3 * (Y=5)
140 PRINT A$ ;S+1;B$ ;A;C$ ;B : NEXT
```

程序说明：

①10～40 行为人机对话和初始化程序。

②50～60 行设置 K 标志。

③70～140 行计算输出结果。其中 90 行求 S，100 行求 Y，110 行确定 P 标志，120、130 行求 A、B 的值。

运行程序，输入

discs＝3

Seats＝2　　则打印如下结果

(1) Disc 1 from 1 to 2

(2) Disc 2 from 1 to 3

(3) Disc 1 from 2 to 3

(4) Disc 3 from 1 to 2

(5) Disc 1 from 3 to 1

(6) Disc 2 from 3 to 2

(7) Disc 1 from 1 to 2

若输入

discs＝3

Seats＝2　　则打印

(1) Disc 1 from 1 to 3

(2) Disc 2 from 1 to 2

(3) Disc 1 from 3 to 2

(4) Disc 3 from 1 to 3

(5) Disc 1 from 2 to 1

(6) Disc 2 from 2 to 3

(7) Disc 1 from 1 to 3

对比两个运行结果可见，N＝3 时，各号金片的移动顺序都是相同的。

若输入

discs＝4

Seats＝3　　则打印

(1)　Disc 1 from 1 to 2

(2)　Disc 2 from 1 to 3

(3)　Disc 1 from 2 to 3

(4) Disc 3 from 1 to 2

(5) Disc 1 from 3 to 1

(6) Disc 2 from 3 to 2

(7) Disc 1 from 1 to 2

(8) Disc 4 from 1 to 3

(9) Disc 1 from 2 to 3

(10) Disc 2 from 2 to 1

(11) Disc 1 from 3 to 1

(12) Disc 3 from 2 to 3

(13) Disc 1 from 1 to 2

(14) Disc 2 from 1 to 3

(15) Disc 1 from 2 to 3

运行结果与表 9—3 相同。

若固定目标针座为 3,引入数组和符号判别式,省略金片号数提示(实际上,只要给出针座的顺序,就可以确定金片号数),可使程序极大地简化。下面再给出一个梵塔搬移的程序供读者分析(见 No. 9—28)。

```
5 REM No. 9—28
10 CLS : INPUT "N=";N : DIM X(N,2),A(8)
20 FOR I=1 TO N : X(I,0)=I : NEXT
30 G=1 : FOR I=1 TO N : G=G * 2 : NEXT
40 A(0)=N : FOR I=1 TO G-1
50 F=E+1+(E>1) * 3 : A=X(A(E),E) : B=X(A(F),F):M=-
   (A>B) * E-(A<B) * F:F=F+E-M
60 PRINT "("I") moves from " M+1 "to" F+1
70 A(F)=A(F)+1 : X(A(F),F)=X(A(M),M) : A(M)=A(M)-1
   : E=E+2+(E>0) * 3 : NEXT
80 END
RUN     ↙键入 3,则打印
(1) moves from 1 to 2
```

(2) moves from 1 to 3

(3) moves from 2 to 3

(4) moves from 1 to 2

(5) moves from 3 to 1

(6) moves from 3 to 2

(7) moves from 1 to 2

运行结果与 No. 9—27 相同。

9.5.2 金叶古塔搬移的动态模拟

程序 No. 9—24 形象直观地模拟了八皇后问题的求解过程。梵塔的搬移过程能否进行动态模拟呢？现在讨论这一问题。

上一节我们在讨论金片的搬移过程时，仅考虑确定某号金片从几号针座移到几号针座，不必考虑金片在针座上的位置，而进行动态模拟，则必须解决原针座上金片的擦除和移入针座上金片的绘制两个问题。首先要确定擦除、绘制金片的位置座标；其次要确定擦除、绘制金片的宽度。

我们规定最小的金片宽度最窄，最大的金片最宽。定义 B 数组、S 数组并通过它们之间的组合分别用于表示原针座、中间针座和目标针座；定义 X 数组记录三个针座的位置；定义 P 数组记录三个针座上的金片数量；定义 A(a,b)数组记录各针座上各金片的宽度，其中 a 记录针座号，b 记录金片个数。编程如下：

```
5 REM No. 9—29
10 INPUT "N=3,4";N : DIM P(3),X(3),A(3,N) : CLS
20 PRINT " * * * "Jin Ye Gu Ta Ban Yi␣ * * * "
30 X(1)=20 : X(2)=29 : X(3)=38 : Y=18 : A(1,1)=19 : B=1 : P
   (1)=N
40 LOCATE 0,18 : FOR A=1 TO 27 : PRINT CHR$(227); : NEXT
50 LOCATE 4,19 : PRINT "A" : LOCATE 13,19 : PRINT "B" : LO-
   CATE 22,19 : PRINT "C"
60 FOR I=2 TO N : A(1,I)=A(1,I-1)-1 : NEXT
```

```
70 FOR I=1 TO N : P=I : GOSUB 190 : NEXT
80 DIM N(20),B(20),F(20),S(20)
90 I=0 : N(1)=N : J=1 : B(1)=1 : F(1)=3 : S(1)=2 : GOSUB
   110
100 LOCATE 0,23 : PRINT "OK!"; : END
110 I=I+1 : IF N(I)=0 THEN I=I-1 : RETURN
120 N(I+1)=N(I)-1 : B(I+1)=B(I) : F(I+1)=S(I) : S(I+1)=
   F(I) : GOSUB 110
130 LOCATE 12,2 : PRINT J : J=J+1 : B=B(I) : F=F(I) : LO-
   CATE 0,22 : PRINT "Press any key……"; : A$=INKEY$(0)
140 LOCATE 11,3 : PRINT CHR$(64+B) "---" CHR$(64+F)
    : A$="  " : P=P(B) : GOSUB 200 : P(F)=P(F)+1
150 A(F,P(F))=A(B,P(B))
160 P(B)=P(B)-1 : B=F : P=P(B) : GOSUB 190
170 N(I+1)=N(I)-1 : B(I+1)=S(I) : F(I+1)=F(I) : S(I+1)=
   B(I) : GOSUB 110
180 I=I-1 : RETURN
190 A$=CHR$(253)
200 FOR W=X(B)-A(B,P) TO X(B)+A(B,P)-32 : LOCATE W,
   Y-P : PRINT A$; : NEXT
210 RETURN
```

运行程序，键入“4，”显示如图 9—36 的画面。

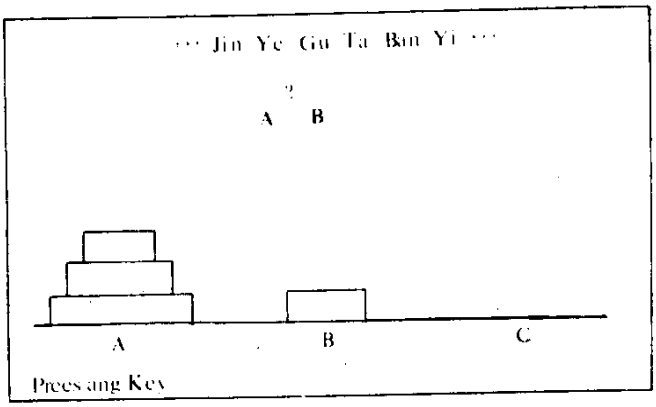


图 9—36　金圳古塔搬移动态模拟画面

按任一键移动一次，直到四片金叶全部移到针座C上，演示结束。

程序说明：

①10～80行为人机对话、初始化程序。其中20行打印标题，40行画针座，50行打印针座号，70行画金片。

②90～180行模拟金片的移动过程，其中140行完成擦除金片；160行完成金片的绘制。

③200～210行为擦、绘金片的子程序。

由于显示屏幕的限制，程序No.9—29只演示N＝3，N＝4的搬移过程，若改变30行的X数组各变量和A(1,1)的数值，可最多演N＝5的搬程过程，若想演示的金叶层数更多，可把三个针座纵向排，则可演示N＝9的搬移过程，留给读者练习。

通过以上的讨论，我们已知道64层的梵塔全部移完需要$2^{64}-1$次。第六章我们曾计算过

$$2^{64}-1=1.8\times10^{19}$$

那么，我们假定贝拿勒斯圣庙的僧侣们一秒钟移动一次的话，则移完梵塔要用5714亿年之多，可见，梵天的预言并非骇人听闻，五千亿年后的世界是个什么模样，谁也难以预料。

9.6 四色猜想的验证

四色猜想最初由英国大学生弗朗西斯·格里思在1852年提出来的，其内容是：在给平面或球面上的任何地图着色时，最多用四种不同颜色就可以把所有相邻的区域分开。这个问题刚提出来时，许多数学家都不屑一顾，觉得它过于简单。但很快就发现这是一个极难证明的问题。在长达一百余年的时间里，一批又一批的杰出的数学家和成千上万个数学爱好者都曾为四色问题的证明而绞尽脑汁，但是都未如愿，因此只能称它为“四色猜想”留于后世。四色猜想是世界公认的近代数学三大难题之一。直到1978年，才由

美国的两位数学家阿佩尔和哈肯通力合作，利用当时最先进的三台 IBM 360 型高速电子计算机，运行了 1200 小时，检验了所有可能的图形组合方式，终于使四色问题获得了完善的证明，从而使“四色猜想”变为“四色定理”。

我们可以使用 F BASIC 语言编程验证四色定理。编程思路是：采用回溯法来处理各区域的染色问题，即用 1～4 号颜色逐一试探，凡是使已涂区域各相邻者的颜色不同就被认可，否则换一种颜色再试，直至使所有区域都涂上合适的颜色。当第一区域使用一种颜色试过后，再换第二种颜色，以后换第三种、第四种。每当试出一组符合要求的解，就把它们打印出来，最终找出所有可能的解，程序清单见 No. 9—30。

```
5 REM No. 9—30
10 CLS : READ N : DIM A(N,N),B(N)
20 FOR I=1 TO N : PRINT I ")";
30 FOR J=1 TO N : READ A(I,J) : NEXT : NEXT : PRINT :
   PRINT
40 FOR I=1 TO 4 : B(1)=I
50 FOR J=2 TO N : C=0
60 C=C+1 : IF C>4 THEN 130
70 K=1
80 IF A(J,K)=1 AND C=B(K) THEN 60
90 K=K+1 : IF K<=N THEN 80
100 B(J)=C : NEXT
110 FOR K=1 TO N : PRINT B(K) "␣"; : B(K)=0 : NEXT
120 PRINT : T=T+1
130 NEXT
140 PRINT : PRINT "TOTAL=" T : END
150 DATA N
160 DATA
.....
```

我们先介绍程序 No. 9—30 的使用方法，然后再介绍程序说明。

假设有一张如图 9—38 的区域图，图中共有六个区域，若再加上底面共有七个区域，要求对区域图中的每一区着色，最多用四种颜色把各区域隔开。

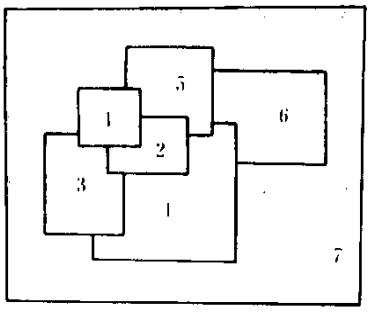


图 9—38

首先对各区域编号(注：编号是任意的)；其次建立数学模型。我们规定两区域相邻记为“1”，两区域相隔记为“0”，区域本身相比记为“0”。根据这一原则，对于图 9—38 可列出各区域对应关系表，见表 9—5。

表 9—5　图 9—38 各区域对应关系表

区域号	1	2	3	4	5	6	7
1	0	1	1	0	1	0	1
2	1	0	1	1	1	1	1
3	1	1	0	1	0	0	1
4	0	1	1	0	1	1	1
5	1	1	0	1	0	1	1
6	0	0	0	1	1	0	1
7	1	0	1	1	1	1	0

表 9—5 的填法如下。(以第一行“0110101”为例)：根据规则，1 区对 1 区记为 0；1 区与 2 区相邻记为 1；1 区与 3 区相邻记为 1；1 区与 4 区相隔记为 0；1 区与 5 区相邻记为 1；1 区与 6 区相隔记为 0；1 区与 7 区相邻记为 1；所以第一行表示为“0110101”。其它各行

类推。

把表 9—5 的数据写入程序 No. 9—30 的 DATA 语句中。首先把 150 行的 N 改为 7，以后各语句行写入一行数据。程序如下：

```
150 DATA 7
160 DATA 0,1,1,0,1,0,1
170 DATA 1,0,1,1,1,1,1
180 DATA 1,1,0,1,0,0,1
190 DATA 0,1,1,0,1,1,1
200 DATA 1,1,0,1,0,1,1
210 DATA 0,0,0,1,1,0,1
220 DATA 1,0,1,1,1,1,0
```

填完 DATA 语句运行程序，即打印出如下结果：

```
1)  2)  3)  4)  5)  6)  7)
1   2   3   1   3   2   4
2   1   3   2   3   1   4
3   1   2   3   2   1   4
TOTAL=3
```

运行结果表明有三种染色方案，现在我们逐一验证其正确性。

设四种颜色分别如下，见图 9—39。

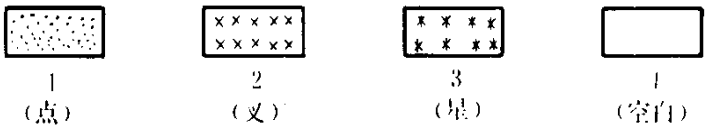


图 9－39　四种颜色代号

对三种组合的染色方案验证见图 9—40(a)、(b)、(c)。

由图 9—40 可见，三种染色方案都是正确的。

程序说明：

①定义两维数组 A(N,N)存放相邻区域的邻接关系。比如 A(1,2)＝1 表示，区域 1 与区域 2 相邻；A(1,4)＝0，表示区域 1 与区域 4 相隔。定义一维数组 B(N)存放各区域的颜色号数，如 B

(6)=2,表示第六区域染第 2 号颜色。

②10～30 行为准备程序。由 10 行读入染色区域的数量,由 30 行读入各区的邻接关系。

③40～130 行完成各区的颜色组合和结果输出,其中由 80 行保证邻区颜色不同,110 行输出结果。

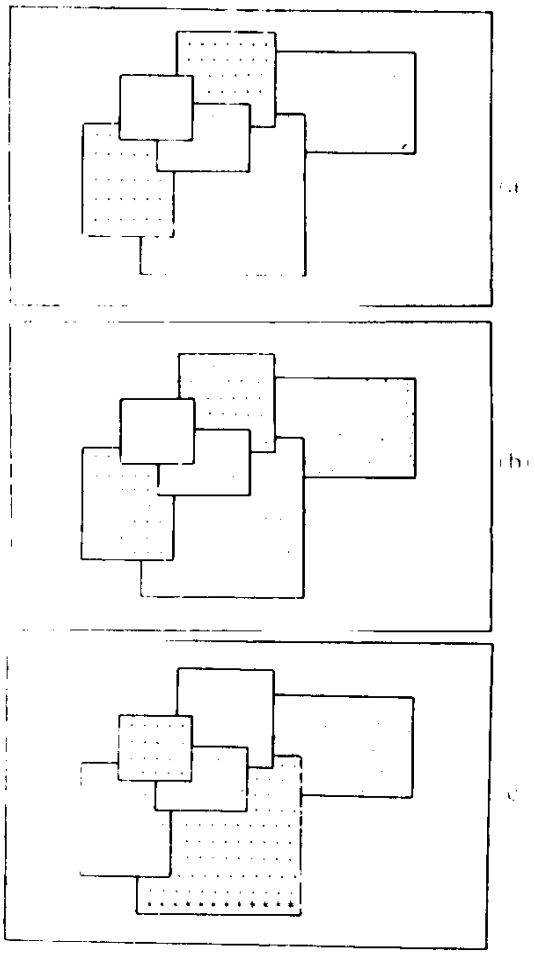


图 9—40

为加深印象,下面再给出两例和运行结果,由读者自己完成数据的输入和验证结果的正确性。

例 9.2 八区域染色见图 9—41。

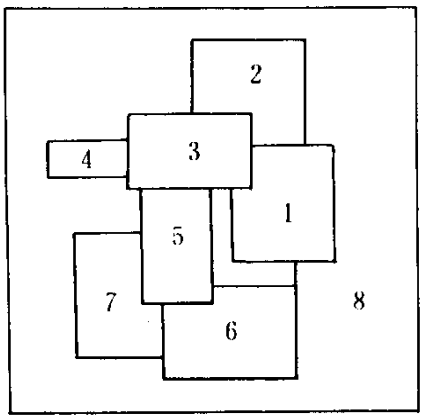


图 9—41

运行结果如下

1)	2)	3)	4)	5)	6)	7)	8)
1	2	3	1	1	2	3	4
2	1	3	1	1	2	3	4
3	1	2	1	1	2	3	4

例 9.3 十二区域染色，见图 9—42。

运行结果如下：

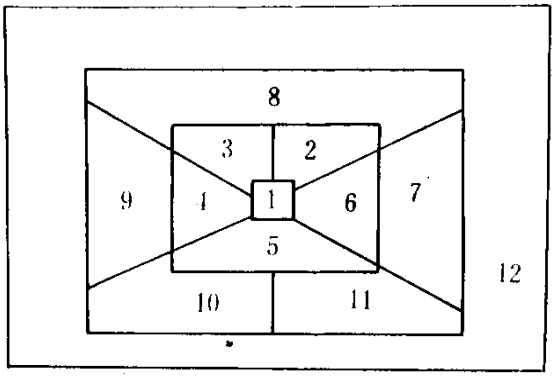


图 9—42

1)	2)	3)	4)	5)	6)	7)	8)	9)	10)	11)	12)
1	2	3	2	3	4	1	4	1	4	2	3
2	1	3	1	3	4	2	4	2	4	1	3

注:输入数据时 150 行的 DATA 语句只输入区域数。

9.7 组合一元钱

9.7.1 问题的提出

有人提出这样一个问题:用一元及以下的人民币组合一元钱共有多少种组合方式。乍看上去,这好象是一个简单的排列组合问题,不值得研究探讨。但仔细分析一下就会发现,它远不是一般的排列组合问题,不但不简单,而且很复杂。由于它用的组合基数分别是 1、2、5、10、20、50 六个数,而这六个数之间又无明显的联系,很难写出解题的公式。若用人工逐个排列组合,则费工费时还不可靠;若用数学方法求解也不是一件容易的事情。因此,最方便有效的方法就是用计算机解决。

9.7.2 最初的方法

那么应如何编写程序呢?我们首先想到的是使用循环语句,采用穷举方法,逐个排列组合,最终求出结果。程序见 No. 9—31。

```
5 REM No. 9—31
10 CLS : PRINT "Please wait……" : K1=2
20 FOR I=0 TO 1 : S=0
30 FOR J=0 TO 5 : FOR K=0 TO 10
40 FOR L=0 TO 20 : FOR M=0 TO 50
50 N=0
60 S=I * 50+J * 20+K * 10+L * 5+M * 2+N
70 IF S<100 THEN 100
80 IF S=100 THEN K1=K1+1 : IF K1 MOD 100=0 THEN PRINT
   "K1=";K1
```

```
90 GOTO 110
100 N=N+1 : IF N<101 THEN 60
110 NEXT : NEXT : NEXT : NEXT : NEXT
120 PRINT "TOTAL=" K1
130 END
```

程序中用 K1 记录排列的方式数，并假定已对 100 及 50+50 两次排列计数，所以对 K1 赋初值 K1=2。

程序 No. 9－31 运行 126 分钟打印

TOTAL=4563

可见，组合方式之多是难以预料的。

程序 No. 9－31 虽然精确地求出了结果，但运行速度太慢。可以算出其总循环次数高达 1400 万次，难怪运算速度如此之慢，有没有提高运行速度的方法呢？回答是肯定的。

9.7.3 采用堆栈技术提高速度

提高速度的一种有效方法是使用堆栈技术。编程思路如下：

1. 把基数 50、20、10、5、2、1 按顺序规定其级别从高到低，本次级别为 50。

2. 假定由 100 组成 100 的次数已记录，所以计数器置 1。

3. 将本级别数进栈，并求进栈数之和。

4. 若和小于 100 则转第 3 步。

5. 若和大于 100 则转第 7 步。

6. 若和等于 100 则计数，并判断结束标志，及时打印结果、停机。

7. 将级别为 1 的数全部退栈，再退栈一次，将本次级别减一个级别，转第 3 步。

8. 结束标志，以最低级别进栈数为 100 时结束。

注：关于堆栈的概念，请参阅有关书籍。

根据上述思路，编程如 No. 9－32。

```
5 REM No. 9—32
10 CLS : DIM K(5),Q(5),R(100) : PRINT "please wait……"
20 K(0)=50 : K(1)=20 : K(2)=10 : K(3)=5 : K(4)=2 : K(5)=1
   : X=1
30 N=N+1 : A=A+K(L) : R(N)=L : Q(L)=Q(L)+1
40 IF A<100 THEN 30
50 IF A>100 THEN 80
60 X=X+1 : IF Q(5)=100 THEN PRINT "TOTAL=" X : END
70 IF X MOD 100=0 THEN PRINT "X=" X
80 IF L=5 THEN A=A-Q(5) : N=N-Q(5) : Q(5)=0
90 L=R(N) : A=A-K(L) : Q(L)=Q(L)-1 : N=N-1 : L=L+1
   : GOTO 40
```

运行程序，87 分钟后打印出相同的运算结果。

相对 No. 9—31，No. 9—32 的运行速度有了大幅度的提高（提高率接近 50%），但是仍然不够理想。分析影响执行速度的原因主要是由于使用数量多、频率高的 1 分币和 2 分币每次都要一个一个地进栈、出栈，并要进行反复地判断，从而造成运算时间的增加。因此，欲进一步提高处理速度，必须对堆栈的算法进行重新设计。

9.7.4 堆栈程序结构的改造

改造思路如下：

1. 定义一个 B(4)数组分别存放 50、20、10、5 四个基数；不再设置存放具体币值的栈，而定义一个 G(4)数组存放已入栈各基数的个数；用变量 J 作为这两个数组的共同指针，初值为 0；用变量 S 表示已入栈币值的和，初值为 0；用 N 累计组合数，初值为 1。

2. 1、2 分币都不再一一入栈、出栈，凡涉及到它们的组合数均用公式法求出。可以验证对于由 50 分～5 分币的任一组合（其值<100），总有如下关系式：

$$N=INT((100-S)/2)+1$$

N 表示对给定的 S 值时由 1 分、2 分币的组合数。

3. 用 INT((100−S)/B(J)算出本级应入栈的个数，并对 G(J)和 S 赋值。

4. 如果 S=100，由计数变量 N 加 1 并退栈。

5. 数组指针 J 加 1，如果 J<4，转第 3 步。

6. 如果 J=4，表示已轮到 2 分币，按上述公式计数后，数组指针向回退，直至相应的 G(J)不为 0，退栈后转第 5 步。在这个过程中，如果 J=−1 就表示运算结束，打印结果后停机。

按照上述思路，编程如 No. 9—33。

```
5 REM No. 9—33
10 DIM B(4),G(4) : N=1
20 FOR I=0 TO 3 : READ B(I) : NEXT
30 DATA 50,20,10,5
40 G(J)=(100−S)/B(J)
50 S=S+G(J) * B(J)
60 IF S<100 THEN 80
70 N=N+1 : G(J)=G(J)−1 : S=S−B(J)
80 J=J+1 : IF J<4 THEN 40
90 N=N+1(100−S)/2+1
100 J=J−1 : IF J=−1 THEN PRINT N : END
110 IF G(J)=0 THEN 100
120 G(J)=G(J)−1 : S=S−B(J) : GOTO 80
RUN ↙
```

运行 10 秒钟，输出结果 4563。

可见，程序 No. 9—32 通过改造后，运行速度比原来提高了 500 余倍。

9.7.5 效率更高的排列组合方法——线性结构程序的优化

在本节的开始，我们讨论了线性结构程序 No. 9--31，该程序运行了 126 分钟才求出结果，结果的计算速度是极为缓慢的。分析造成速度慢的原因，主要是循环次数太多。致使循环次数多的原因

则是因各层循环的终值选择不合理，造成了无效循环次数增加。如，当 50 分币取 1 时(I=1)，20 分币最多只能取 2 个(J=2)，J=3、J=4、J=5 三次都是无效循环；同样，当 I=1，J=2 时，10 分币最多只能取 1 个(K=1)，K=2～K=10 的 9 次循环都是无效的。可见，无效循环量是极大的。因此，每个循环变量的终值都应以外层循环的取值为参考，如 J 的循环终值可取 J=(100−I*50)/20、K 的循环终值可取 K=(100−I*50−J*20)/10，同理 L、M、N 的循环终值依次为：

L=(100−I*50−J*20−K*10)/5

M=(100−I*50−J*20−K*10−L*5)/2

N=100−I*50−J*20−K*10−L*5−M*2

把上述各循环变量的终值代入 No. 9—31 即得程序 No. 9—34。

```
5 REM No. 9—34
10 K1=2 : DIM A(4)
20 FOR I=0 TO 1 : S=0 : A(0)=I*50
30 FOR J=0 TO (100−A(0))/20 : A(1)=A(0)+J*20
40 FOR K=0 TO (100−A(1))/10 : A(2)=A(1)+K*10
50 FOR L=0 TO (100−A(2))/5 : A(3)=A(2)+L*5
60 FOR M=0 TO (100−A(3)/2 : A(4)=A(3)+M*2
70 N=0
80 S=A(4)+N
90 IF S<100 THEN 120
100 IF S=100 THEN K1=K1+1
110 GOTO 130
120 N=N+1 : IF N<100−A(4)+1 THEN 80
130 NEXT : NEXT : NEXT : NEXT : NEXT
140 PRINT "TOTAL=" K1 : END
```

运行程序，21 分钟后打印出相同的结果，与 No. 9—31 相比，计算速度提高了 83%。程序的优化取得了可喜的一步。

程序 No. 9—34 的执行速度仍然较慢，分析程序发现 N 的循

环中仍存在着大量的无效循环，(70～120 行)。事实上，对于确定的 I、J、K、L、M 只有一个唯一的 N 与之对应，也就是说在 N＝0 至 100－A(4)之中的诸多次判断中，只有一次有效。显然删除 N 循环可使运算速度进一步加快，见程序 No. 9—35：

```
5 REM No. 9—35
10 K1=2 : DIM A(4)
20 FOR I=0 TO 1 : S=0 : A(0)=I * 50
30 FOR J=0 TO (100－A(0))/20 : A(1)=A(0)+J * 20
40 FOR K=0 TO (100－A(1))/10 : A(2)=A(1)+K * 10
50 FOR L=0 TO (100－A(2))/5 : A(3)=A(2)+L * 5
60 FOR M=0 TO (100－A(3))/2 : A(4)=A(3)+M * 2
70 IF A(4)<=100 THEN K1=K1+1
80 NEXT : NEXT : NEXT : NEXT : NEXT
90 PRINT "TOTAL=" K1 : END
```

程序中由 70 行记录给定的 I、J、K、L、M 值的排列次数。可以算出，现在的循环次数为 4561 次，仅占 No. 9—31 的 0. 03%。

运行程序，41 秒钟后打印 TOTAL＝4563 的结果。

我们若引用一分币、二分币的计算公式，可再减少一层循环，使计算速度进一步提高。见 No. 9—36。

```
5 REM No. 9—36
10 K1=2 : DIM A(3)
20 FOR I=0 TO 1 : A(1)=I * 50
30 FOR J=0 TO (100－A(1))/20 : A(2)=A(1)+20 * J
40 FOR K=0 TO (100－A(2))/10 : A(3)=A(2)+10 * K
50 FOR L=0 TO (100－A(3))/5
60 K1=K1+(100－A(3)－L * 5)/2+1
70 NEXT : NEXT : NEXT : NEXT
80 PRINT "TOTAL=" K1 : END
```

程序中 60 行直接计算 1 分、2 分币的排列次数。可以算出 No. 9－36 的循环次数仅有 342 次。

运行程序，仅用 2.5 秒钟就打印出结果

TOTAL＝4563

程序 No. 9－36 应是解决“排列组合”一元钱问题的最简、效率最高的程序了。你还有更快的方法吗？请读者试一试。

9.8 趣味取球

1. 题目：现有 41 个球，游戏者和电脑交替从中取球，规定每次只能取一个、二个或三个，谁取到最后一个谁输。

2. 原理：这是数学上的平衡问题。每人一次至少取一个球，至多取三个。无论先取的人采取什么对策，后者总可保证一轮共取四个。那么，只要留给对手 4n＋1(n＝0,1,2,3,……)个就能取胜，程序清单见 No. 9－37。

```
5 REM No. 9—37
10 CLS : N=41
20 PRINT "Ni xian qu ma? (Y/N)"; : INPUT X$ : IF X$ ="Y"
   THEN 100
30 IF N=1 THEN T=1 : GOTO 160
40 CLS : PRINT "Wo qu 1 ge ("N+X "－1－" X"＝" N－1 ")"
50 PRINT : PRINT : PRINT "Ni cong" N－1 "Zhong qu(1、2、3)";
60 INPUT X : IF X>3 OR X<1 GOTO 180
70 IF X=3 THEN N=N－1 : GOTO 130
80 N=N－X－1 : IF N－1 MOD 4=0 THEN 30
90 PRINT "Wo qu" 3－X "ge" : N=N－3+X : GOTO 150
100 PRINT : PRINT : PRINT "Ni cong "N" zhong qu(1,2,3)"
110 INPUT X : IF X>3 OR X<1 GOTO 180
120 IF N－X=1 THEN T=1 : GOTO 160
130 CLS : PRINT "Wo qu" 4－X "␣ge(" N "－" X "－" 4－X "＝" N
    －4 ")"
```

```
140 N=N-4
150 IF N>1 THEN 100
160 IF T=1 THEN CLS : PRINT "Ni ying le!" : END
170 PRINT : PRINT : PRINT "Ni shu le!" : END
180 CLS : PRINT "Ni fan gui le!!!" : IF X$ ="Y" GOTO 100
190 GOTO 50
```

运行程序，清屏，显示

Ni xian qu ma? (Y/N)? □ （询问你先取吗?）

若先取按“Y”键，否则按“N”键。若先取，按“Y”键，则显示

Ni cong 41 zhong qu(1,2,3)? □

（你从 41 中取球，每次取 1，2，3 个）。若取 3 个则按数字键“3”。回车后，清屏，显示

Wo qu 1 ge (41-3-1=37)

Ni cong 37 zhong qu(1,2,3)? □

（我取 1 个）

（你从 37 中取(1,2,3)）

就这样一直取下去，最后有

Ni cong 5 zhong qu(1,2,3)? □

这时，你无论取几都输。如取 1，则显示

Wo qu 3 ge (5-1-3=1)

Ni shu le! （你输了）

通过游戏发现，若你先取，无论取几，电脑都取相对于 4 的补数，留给你从 4n+1 个中取，所以，你必输无疑。

你可能想到让计算机先取。现在，我们试一下：游戏开始，当询问“你先取吗?”时，键入“N”，则清屏，显示

Wo qu 1 ge (41-1=40)

Ni cong 40 zhong qu(1,2,3)

（我取 1 个，你从 40 中取）

若我们取 2 个，则键入“2”，回车后，显示

Wo qu 1 ge

Ni cong 37 zhong qu (1,2,3)

如此，取到最后仍然是你输。

那么，怎样能赢呢？请朋友动动脑筋。

程序说明：

①20 行人机对话，决定谁先取球。

②30 行是游戏者胜的转移出口。

③40～90 行为电脑先取的处理程序。

④100～150 行为游戏者先取的处理程序。

⑤160～190 行为输赢判定，犯规提示和处理程序。

当你掌握了该游戏的取胜规律后，可把 10 行改为

10 CLS：INPUT "N＝";N

然后运行程序，试一试对于任意给定的 N 能否局局得胜。再把 10 行改为

10 CLS：N＝RND(20)＋30

随机给出 N 值，看是否仍能取胜。

9.9 有趣的整数

9.9.1 有趣的整数

自然数中存在着一类这样的数：当它乘以数字 2～9 中任一个时，所得之积的各位和与原数的各位和相等。如 18：

18×2＝36，　　3＋6＝1＋8＝9

18×3＝54　　5＋4＝1＋8＝9

……

18×9＝162，　　1＋6＋2＝1＋8＝9

再如 45：

45×2＝90，　　9＋0＝4＋5＝9

……

45×9=405，　　4+0+5=4+5=9

再如 594：

594×2=1188，　　1+1+8+8=5+9+4=18

594×3=1782，　　1+7+8+2=5+9+4=18

594×9=5346，　　5+3+4+6=5+9+4=18

试编程找出这类有趣的整数。

解决这一问题，编程思路很简单：即令 N 等于自然数 1，首先求出 N 的各位和为 A，然后令 N 分别乘以 2～9，再对乘积逐位求和得出 B，若每一个 B 都与 A 相等，则 N 满足条件，打印输出，否则 N=N+1，重复以上操作。

根据上述思路，很容易写出如下程序：

```
5 REM No. 9—38
10 N=1
20 N$=STR$(N):A=0
30 FOR I=1 TO LEN(N$)
40 A=A+VAL(MID$(N$,I,1)):NEXT
50 J=2
60 X=N*J:X$=STR$(X):B=0
70 FOR K=1 TO LEN(X$)
80 B=B+VAL(MID$(X$,K,1)):NEXT
90 IF B<>A THEN 120
100 J=J+1:IF J<=9 THEN 60
110 PRINT N;
120 N=N+1:GOTO 20
```

运行程序，可打印出结果

9　18　45　90　99　180……

通过运行可见，程序 No. 9—38 的计算速度太慢，影响运算速度的主要原因是：采用穷举法查找，无效循环太多。有没有提高速度的方法呢？

我们从运行结果中可以找出如下规律：

①这类数都能被 9 整除。

②在自然数的序列中，这类数的取值跳跃性较大、且步长不一，但不论其长短，也都是 9 的倍数。因此，我们可从 N＝9 开始，每次加 9 进行检查，打印满足条件的 N。这样可使程序的运算速度提高近 90%。

另外，这类整数能够被 9 除的性质，我们也可以用数学方法证明。

设 ab 是一个两位的正整数，如果

$$\begin{cases} ab\times 2==a1b1c1 & (1) \\ ab\times 3=a2b2c2 & (2) \end{cases}$$

且有 a1＋b1＋c1＝a2＋b2＋c2＝a＋b

则 ab 能被 9 整除。

证明：

(1)(2)式减(1)式得

$$\begin{aligned} ab &= a2b2c2-a1b1c1 \\ &= 100(a2-a1)+10(b2-b1)+(c2-c1) \end{aligned} \quad (3)$$

(2)由于

$$a2+b2+c2=a1+b1+c1$$

所以得

$$a2-a1=(b1-b2)+(c1-c2) \quad (4)$$

(3)把(4)式代入(3)式右边得

$$\begin{aligned} ab &= 90(b1-b2)+99(c1-c2) \\ &= 9[10(b1-b2)+11(c1-c2)] \end{aligned}$$

所以 ab 能被 9 整除。

读者可试证明三位数、四位数的情况。

根据上述的分析论证，编写程序见 No. 9—39。

```
5 REM No. 9—39
10 CLS : INPUT "C="；C : PRINT "C=" C
20 I=I+9 : N=I : GOSUB 100 : B=A : K=2
```

```
30 N=I*K:GOSUB 100:IF A<>B THEN 70
40 K=K+1:IF K<10 THEN 30
50 PRINT I;
60 M=M+1
70 IF I<=C THEN 20
80 PRINT:PRINT:PRINT "M=" M
90 END
100 N$=STR$(N):A=0
110 FOR J=1 TO LEN(N$)
120 A=A+VAL(MID$(N$,J,1))
130 NEXT:RETURN
```

程序 No. 9—39 对 No. 9—38 进行了改造，除把步长改为 9 外，还把 30～40、70～80 两段求各位和的程序改为通用求和子程序，从而使程序简化。另外 No. 9—39 中还加入了求数范围控制变量 C，和数据记录变量 M。

运行程序，要求用户输入查找数据的范围，键入 3000，则打印出如下结果

C=3000

9	18	45	90	99	180	198
297	396	450	495	549	594	693
792	819	891	900	909	918	945
990	999	1089	1188	1287	1386	1485
1584	1683	1728	1782	1800	1809	1818
1908	1980	1998	2079	2178	2376	2475
2574	2673	2718	2745	2907	2970	2997

M=49

共找出 3000 以内的 49 个数具有上述性质。

9.9.2 有趣整数的推广

以上讨论的有趣整数，仅限于乘以 2～9 八个数，若把它推广到乘以任意个自然数仍具有上述性质，那么，这类数将是哪些？

欲达到这一目的，我们只要在 No. 9—39 中引入一个控制变量 D 即可实现。见 No. 9—40

```
5 REM No. 9—40
10 CLS : INPUT "C、D=";C,D : PRINT "C=";C : PRINT "D="D
20 I=I+9 : N=I : GOSUB 100 : B=A : K=2
30 N=I * K : GOSUB 100 : IF A<>B THEN 70
40 K=K+1 : IF K<D THEN 30
50 PRINT I;
60 M=M+1
70 IF I<=C THEN 20
80 PRINT : PRINT : PRINT "M=" M
90 END
100 N$ =STR$ (N) : A=0
110 FOR J=1 TO LEN(N$ )
120 A=A+VAL(MID$ (N$ ,J,1)
130 NEXT : RETURN
```

RUN ↙ 显示

C,D=? □ 键入 1000,20 则打印

C=1000

D=20

99 198 297 396 495 594

693 792 990 999

M=10

我们可以验证一下结果的正确性，以 495 为例：

495×2=990， 9+9+0=4+9+5=18

495×13=6435， 6+4+3+5=4+9+5=18

495×18=8910， 8+9+1+0=4+9+5=18

495×20=9990， 9+9+9+0=4+9+5=18

可见 495 是满足要求的数。

若运行程序键入“500,50”，则打印

C=500
D=50
99　198　396　495
M=4
若键入“300,100”,则打印
C=300
D=100
99　198
M=2
读者可验证这些结果的正确性。

9.10　有趣的余数问题

已知某个自然数分别被一些自然数除后所得的余数求原数,这就是我国从古至今流传很广的余数问题。它在我国有许多名称,“韩信点兵”就是余数问题中的一个例题。我国古代人还独创了这类问题的解法,就是著名的“中国余数定理”。

在第三章介绍 F BASIC 的基本语句时曾涉及到余数问题,以“韩信点兵”为例给出了推广型的解题程序 No. 3—30。该程序采用穷举法,对所有除数的最小公倍数以内的数逐个验算求出满足条件的解,程序的运行速度较慢。而且只能求除数为三个时的解,若除数增加,还要修改程序中的 3 条语句,故程序的通用性不强。如求“一篮红枣,若将它们分成三颗一组,余一颗;分成五颗一组,余两颗;分成七颗一组余三颗。求红枣有多少颗”。这与“韩信点兵”问题相似,可直接用 No. 3—30 求得答案为 52 颗。但对于爱因斯坦做过的数学题:“一条阶梯,若每步跨 2 阶剩 1 阶;每步跨 3 阶剩 2 阶;每步跨 5 阶剩 4 阶;每步跨 6 阶剩 5 阶;每步跨 7 阶,则一阶不剩。问这条阶梯多少阶?”对于这一问题,则要修改程序才能运算。再如:“有物不知其数,九九数之剩二;八八数之剩三;七七数之

剩四；六六数之剩五；五五数之剩四；四四数之剩三；三三数之剩二；二二数之剩一。问物若何？”。这一题若用程序 No. 9—30 求解则运行时间更长。

实际上，上述所列的问题都有无穷个解，一般只要求出其中最小的解。现在，我们这样构思编程思路：(以阶梯为例)

首先在 1 的基础上每次累加 2，直到被 3 除余 2 时，再在此基础上每次累加 2 与 3 的最小公倍数，直到被 5 除余以 4 时，又在此基础上累加 2、3、5 的最小公倍数，找出被 6 除余 5 的数；然后再累加 2、3、5、6 的最小公倍数，找出整除 7 的数即为答案；最后，输出结果，并给出该条件下所有解的通项表达式。

根据上述编程思路编写出余数问题的通用程序如 No. 9—41。

```
5 REM No. 9—41
10 CLS : READ I,X
20 READ J : IF J<0 THEN 90
30 READ Y
40 IF X MOD J=Y THEN 60
50 X=X+I : GOTO 40
60 S=I : T=J
70 R=S MOD T : IF R<>0 THEN S=T : T=R : GOTO 70
80 I=I * J/T : GOTO 20
90 PRINT "X min=" X
100 PRINT "X=KX" I "+" X
110 DATA 2,1,3,2,5,4,6,5,7,0,-1
120 END
```

程序中 40～50 行求出满足下一个除数条件的数；60～80 行求出 I、J 的最小公倍数。

运行程序，立即打印出如下结果：

$X_{min}=119$

$X=K\times 210+119$

若把 110 行改为

```
110 DATA 9,2,8,3,7,4,6,5,5,4,4,3,3,2,2,1,-1
```

运行程序则求出问题 3 中的答案：

$X_{min}=1019$

$X=K\times 2520+1019$

即该物最少有 1019 个。

朋友们可能觉得程序 No. 9—41 使用不方便，希望删去 DATA 语句的输入，采用人机对话的方式输入数据。程序 No. 9—42 可满足这一要求。

```
5 REM No. 9—42
10 CLS : INPUT "Chu shu=";I : INPUT "Yu shu=";X
20 INPUT "Xia yi ge Chu shu=";J : IF J<0 THEN 100
30 INPUT "Xia yi ge Yu shu=";Y
40 IF X MOD J=Y THEN 60
50 X=X+I : GOTO 40
60 S=I : T=J
70 R=S MOD T
80 IF R<>0 THEN S=T : T=R : GOTO 70
90 I=I * J/T : GOTO 20
100 PRINT "Xmin=" X
110 PRINT "X=K" CHR $ (181);I "+" X
120 END
```

现在我们计算红枣问题：

运行程序，屏幕显示如下提示：

Chu shu=? □ 键入“3” ↙

Yu shu=? □ 键入“1” ↙

Xia yi ge Chu shu=? □ 键入“5” ↙

Xia yi ge Chu shu=? □ 键入“2” ↙

Xia yi ge Chu shu=? □ 键入“7” ↙

Xia yi ge Chu shu=? □ 键入“3” ↙

Xia yi ge Chu shu＝?□ （现在数据已经输入完毕，故键入结束符号“－1”）则立即打印

$X_{min}=52$

$X=K\times105+52$

通过运行程序可见，程序 No. 9—42 的运行速度是极快的，人机对话的过程就是运算的过程，一直是电脑在等待用户的输入。

对于阶梯问题，若根据提示依次输入

“2，1，3，2，5，4，6，5，7，0，－1”

也立即打印出结果。

有物不知其数的问题，依次输入

“9，2，8，3，7，4，6，5，5，4，4，3，3，2，2，1，－1”，也立即打印出相同的结果。

9.11 万年月历

目前，月历已比比皆是。大到悬于壁上的挂历，小到置于案头的台历，还有可放于钱包、证夹和作为书签使用的袖珍年历，种类繁多，不胜枚举。但这些年历都有一个共同特点，就是只刊登当年的（个别的月历有两年的），每年要重新编印一次。这样做，方便于日常使用，但不适于研究历史。本章第一节曾介绍了电脑测星期的程序 No. 9—4。该程序虽然使用可靠，但每运行一次只能求出某一天是星期几，而且没有记忆。若能用电脑打印出任意年、任意月的月历，将会给使用和查阅带来极大的方便。

关于打印月历的程序，PC 机、CEC－I 机都有许多介绍，但这些程序有的使用浮点数计算，有的占用高分辨图形页、有的使用 TAB 函数定位且一行占用 30 个字符以上等等，都不适用于 F BASIC。

笔者从 F BASIC 的特点出发，通过把数字转化为字符的处

理，实现了月历的格式打印，编写了通用万年月历程序 No. 9—43。使用该程序，可打印出任意年任意月的日期、星期对照表。使用极为方便。

```
5 REM No. 9—43
10 CLS : DIM M(12) : INPUT "This Year=";Y : A$="* * * * * * * * * * * *"
20 INPUT "What day of the week is New Year's Day";DD : D=-DD : S=0 : CLS
30 FOR N=0 TO 12 : READ M(N) : NEXT
40 FOR N=1 TO 12 : PRINT
50 S=S+M(N-1) : PRINT Y"======"N"======" : PRINT
60 PRINT "SU M1 T2 W3 T4 F5 S6" : PRINT A$
70 W=1
80 PRINT : G=1
90 D=D+1 : D2=D-S : IF D2<=0 THEN D$="  " : GOTO 120
100 IF D2>M(N) THEN 160
110 D$=STR$(D2) : IF LEN(D$)=2 THEN D$="  "+D$
120 PRINT D$"  ";
130 G=G+1 : IF G<=7 THEN 90
140 IF D2=M(N) THEN 170
150 W=W+1 : IF W<=6 THEN 80
160 D=D-G
170 PRINT : PRINT A$ : B$=INKEY$(0) : NEXT
180 DATA 0,31,28,31,30,31,30,31,31,30,31,30,31
```

程序使用说明：

①运行程序后，看到"This Year="?□的提示后，键入欲查的年份。如 1992、1901 等。

②看到"What day of the week is New Year's Day?"(元旦是星期几?)时，要键入查询年元旦是星期几，星期一～星期六键入"1

～6”，星期天键入“0”。

③首先打印一月份的月历，按任一键，打印下一月的月历，直至结束。

④程序中 180 行 DATA 语句中的数据是按平年给出的，若查询年为闰年，需将 180 行第三个数据“28”改为“29”。

现在，我们欲打印 1992 年的月历，由于 1992 年是闰年，故把 180 行的“28”改为“29”。然后计算 1992 年 1 月 1 日是星期几。计算星期的公式如下：设查询年号为 Y，星期数为 W，则

$$\begin{cases} Y1=Y-1 \\ S=Y1+INT(Y1/4)-INT(Y1/100)+INT(Y1/400)+1 \\ W=S\ MOD\ 7 \end{cases}$$

利用上式可求出 1992 年的元旦是星期三。

运行程序，清屏。显示：

```
This year=? □   键入 1992      显示
What day of the week is New Year's
Day ? □         键入  3    ↙   则打印
1992   ======1======
SU     M1     T2     W3     T4     F5     S6
* * * * * * * * * * * * * * * * * * * * * *
                     1      2      3      4
5      6      7      8      9      10     11
12     13     14     15     16     17     18
19     20     21     22     23     24     25
26     27     28     29     30     31
* * * * * * * * * * * * * * * * * * * * * *
□     (闪烁光标，等待用户按任一键)
```

按任一键后打印 2 月份的月历。

```
1992   ======2======
SU     M1     T2     W3     T4     F5     S6
* * * * * * * * * * * * * * * * * * * * * *
```

```
                                    1
2       3       4       5       6       7       8
9       10      11      12      13      14      15
16      17      18      19      20      21      22
23      24      25      26      27      28      29
* * * * * * * * * * * * * * * * * * * * *
```

□

按任一键后继续打印下一月份的月历。

若欲查询1993年的月历，则把180行的第三个数改为28，计算出1993年的元旦是星期五，运行程序，键入1993和5，则打印

```
1993  =====1=====
SU      M1      T2      W3      T4      F5      S6
* * * * * * * * * * * * * * * * * * * * *
                                        1       2
3       4       5       6       7       8       9
10      11      12      13      14      15      16
17      18      19      20      21      22      23
24      25      26      27      28      29      30
31
* * * * * * * * * * * * * * * * * * * * *

1993  =====2=====
SU      M1      T2      W3      T4      F5      S6
* * * * * * * * * * * * * * * * * * * * *
        1       2       3       4       5       6
7       8       9       10      11      12      13
14      15      16      17      18      19      20
21      22      23      24      25      26      27
28
* * * * * * * * * * * * * * * * * * * * *
```

程序说明：

①10～30 行为人机对话和初始化程序。

②40、170 行控制打印 1～12 个月的月历。

③50～60 行打印每月的标题。

④70、150 行控制打印日期换行。

⑤80～130 行打印日期。其中 90 行的 D,D2,S 控制每月日期的空格数;100 行控制本月打印结束;110 行使日期按格式打印;

⑥160 行为下一月份的开始空格作准备。

⑦保证打印格式化是通过 90、110 两行实现的。当日期为单位数时令其前面加一空格凑成两位数的形式。另外,在 F BASIC 中,当数值通过 STR $ 函数转换为字符串时,都要在前面加一个空格。所以,每个字符串的长度都是 3。再通过 120 行空一格,从而使每个日期都占四个字符位置,保证了打印格式。

⑧为使程序简短,No. 9—43 删去了求星期语句。如求完美,读者可将 No. 9—4 作为子程序合并到 No. 9—43 中。只要修改 No. 9—4 中的个别语句和变量,令 No. 9—43 中 20 行的 D＝S MOD 7,并加一句闰年的处理语句,如

```
IF Y MOD 4＝0 AND Y MOD 100＜＞0 AND Y MOD 400＜＞0
THEN M(2)＝29
```

就可使 No. 9—43 变为更加方便的月历程序,使用时只要输入年份就可打印月历。

9.12 趣味程序——寻宝

程序运行后,屏幕上画出 10×10 的棋盘,棋盘共有 100 个方格,指示器位于棋盘的中央,随着一声蜂鸣,在棋盘的右边或下边闪烁一面小旗,以指示本局宝贝所在的行(或列)。游戏者要记住这一闪烁小旗的位置,用四个方向键控制指示器的移动。当把指示器

移到你认为宝贝所在的方格后，按空格键挖宝。若挖出宝贝则在该方格显示宝贝（苹果），蜂鸣 10 响进行下一局，同时在屏幕的右边打印得分；若没挖出宝贝，则蜂鸣一声，小旗又闪烁一次，再次提示宝贝所在的行（或列）。游戏者必须在固定的行（或列）的 10 个方格中，五次以内挖出宝贝可得满分（第一次挖出得 50、第二次挖出得 40、第五次挖出得 10 分。）；若在第六次挖出则不得分；若在六次以后挖出则得负分。欲结束游戏，按“Esc”键，则清屏，打印总分，挖宝次数和每次所得的平均分数。

与你的朋友比赛一下，看谁得的分数最高。程序清单见 No. 9—44。

```
5 REM No. 9—44
10 VIEW
20 DEF MOVE(0)=SPRITE(10,0,1,1,0,1) : POSITION 0,11 * 8+
   12,11 * 8+20 : MOVE 0
30 SPRITE ON : GOSUB 270
40 LOCATE 0,22 : A$ =INKEY$ (0) : D=ASC(A$ )
50 IF D=27 THEN 340  (结束)
60 IF D=32 THEN 190  (挖宝)
70 IF D<28 OR D>31 THEN 40
80 X1=2 * (D=29)-2 * (D=28)
90 Y1=2 * (D=30)-2 * (D=31)
100 SX=(XPOS(0)-12)/8+X1 : SY=(YPOS(0)-20)/8+Y1
110 X=SX : Y=SY : GOSUB 140 : SX=X : SY=Y
120 POSITION 0,SX * 8+12,SY * 8+20 : MOVE 0
130 GOTO 40
140 IF X>19 THEN X=1
150 IF X<1 THEN X=19
160 IF Y>19 THEN Y=1
170 IF Y<1 THEN Y=19
180 RETURN
```

```
190 SX=(XPOS(0)-12)/8 : SY=(YPOS(0)-20)/8
200 IF SX<>A OR SY<>B THEN 250
210 SPRITE OFF : LOCATE A,B : PRINT CHR$(207) : LOCATE
    TX,TY : PRINT CHR$(199)
220 FOR I=1 TO 10 : BEEP : NEXT
230 CJ=CJ+J*10 : LOCATE 22,3 : PRINT "␣␣␣␣␣" : LO-
    CATE 22,3 : PRINT CJ
240 LOCATE A,B : PRINT "␣" : LOCATE TX,TY : PRINT "␣"
    : GOTO 30
250 J=J-1 : BEEP
260 GOSUB 310 : GOTO 40
270 J=5 : T=T+1 : A=RND(10)*2+1 : B=RND(10)*2+1
280 S=RND(2)+1 : IF S=1 THEN TX=A : TY=21 : GOSUB 310
290 IF S=2 THEN TX=21 : TY=B : GOSUB 310
300 RETURN
310 LOCATE TX,TY : PRINT CHR$(199) : PAUSE 20
320 LOCATE TX,TY : PRINT "␣"
330 RETURN
340 SPRITE OFF : CLS : PRINT "Your SCORE";CJ
350 PRINT "You play time";T
360 PRINT "Your ability" CJ/T "/Time"
370 END
```

程序输入完毕，按图 9—43 键控绘图。

程序说明：

①10～130 行为主程序。其中，20 行定义 10 号卡通爆炸为指示器；30 行调宝贝座标产生子程序，随机产生宝贝座标；40 行读键盘，50 行为结束转移程序（按 Esc 键，D=27）；60 行为按空格键挖宝的处理；70 行封锁除 Esc、空格、方向键外的其它键；80～120 行完成指示器的移动。

②140～180 行为出界处理程序，保证指示器在棋盘中移动。

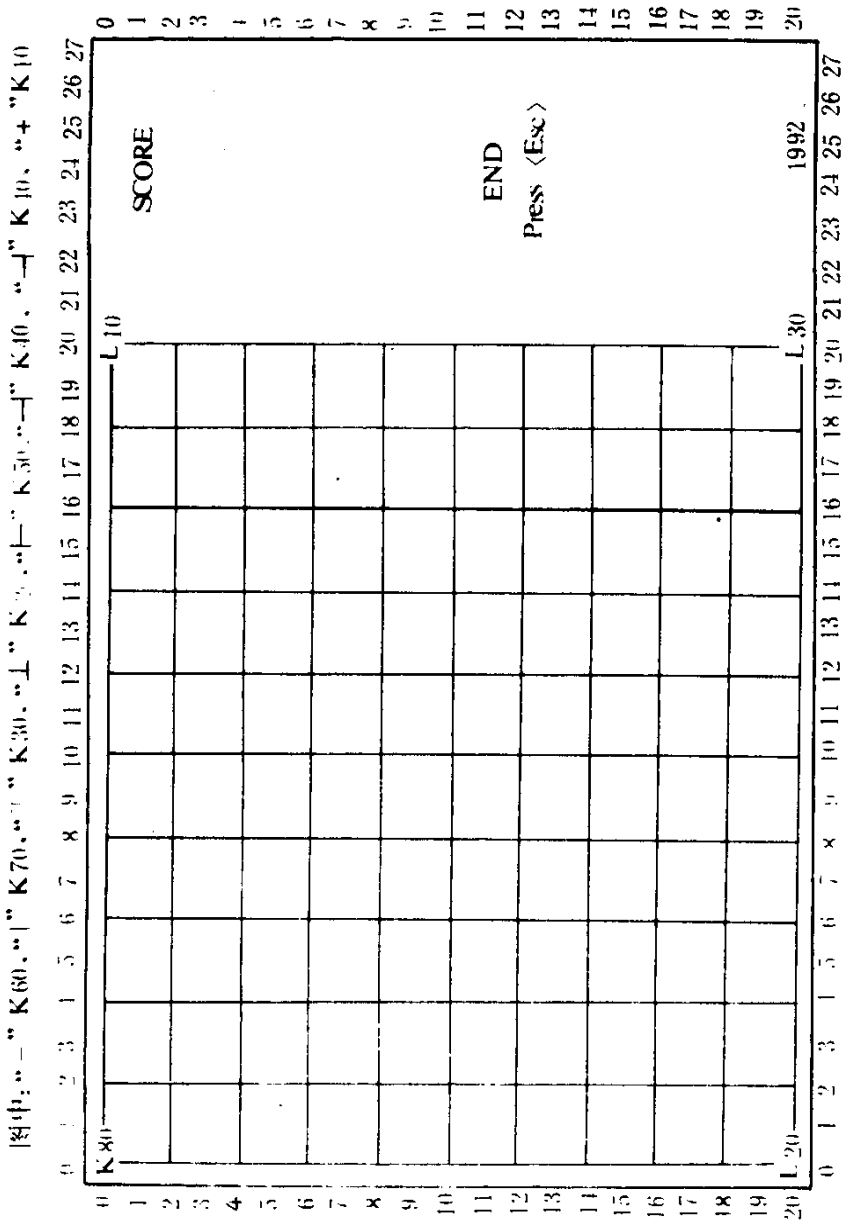


图 9—43

③190～260 行为挖宝处理程序。其中 210～240 行为挖出宝贝的处理(即猜对);250～260 行为未挖到宝贝的处理。由 190～200 行判断是否挖出宝贝。若挖出宝贝,则令指示器消失,在指示器处打印宝贝,同时在宝贝所在的行(列)外显示小旗,鸣响 10 次,打印得分;若没挖出宝贝则调 310 行小旗显示子程序。

④270～300 行,每局游戏开始时,随机产生宝贝的座标,并由 280、290 行决定小旗提示标志显示的座标。

⑤310～330 行令提示标志小旗闪烁一下。改变 PAUSE 语句的数值可改变小旗显示的时间。

⑥340～370 行为结束处理程序。打印游戏者的总分,游戏局数和每局的平均成绩。

⑦欲增加难度(避免游戏者在固定的行、列逐格试验),可在没挖到宝贝后,改变一次宝贝的 Y 座标,改变原则为,若小旗在固定的行显示则改变宝贝的 Y 座标;否则改变 X 座标。

程序 No. 9—44 趣味性较强,稍加改造就可变为较好的智力游戏。有兴趣的朋友不妨一试。

第十章　F BASIC 语言的程序移植技巧

电子计算机诞生以来，应用最广泛的计算机高级语言就是BASIC 语言。几十年来，经过广大计算机工作者、爱好者的共同努力，用 BASIC 语言编写了丰富的应用软件和大量的实用程序。目前，它们仍在各自的机器上发挥着巨大的作用。

近年来，F BASIC 语言初露头角，蹒蹒跚跚地走进了 BASIC 大家族。不少朋友希望把已有的成熟软件移植到 F BASIC 中，以充分发挥游戏电脑的作用。但是，由于 F BASIC 与其它 BASIC 语言间的种种差异造成了程序移植的困难，致使尝试多次，成功率不高。

实际上，程序的移植是一项难度较高的技术工作。它不仅要求操作者全面、深入地掌握移植目标程序语言（即 F BASIC 语言）的所有指令、语法、特点及编程技巧；还要熟悉、了解移植原始程序（被移植的程序）的语言特点及编程技巧。只有掌握了“目标、原始”两种语言之间的联系和差异，才能保证程序的移植工作顺利成功。所以，对于初学电脑游戏机的朋友来说，首先模仿编程，熟悉指令；然后是动手设计融会贯通；最后是深入研究，掌握精髓。在对 F BASIC 语言有了深入的理解、领会，掌握了各种编程技巧以后，再考虑着手程序的移植。只有这样，才能少走弯路，一蹴而就，获得成功。否则，暴虎冯河、急于求成，往往事倍功半，欲速不达。这也正是把本章安排在最后的原因。

为帮助对 F BASIC 语言有更深一步追求的朋友们学习程序

的移植，本章拟从四个方面简要介绍程序移植的方法、步骤。最后通过几个移植示例，具体介绍程序移植中各种情况的处理方法。

由于BASIC语言种类繁多，特点各异，不可能面面具到。本文主要以八位普及型学习机中应用最广泛的APPLE机、中华学习机的CEC－BASIC为例介绍移植方法。最后再列出几个PC机上程序的移植示例。读者只要掌握了程序移植的基本方法、步骤，再移植其它BASIC语言的程序时定会触类旁通，水到渠成。

10.1 程序移植前的准备工作

10.1.1 熟悉CEC－BASIC的基本特点

1.显示模式：

CEC－I有三种显示模式：

(1)文本模式

(2)低分辨图形模式

(3)高分辨图形模式

·在文本模式下有两种显示方式：

①西文方式：字符构成5×7点阵

每场显示40×24个字符

②中文方式：字符构成16×16点阵(汉字)

8×16点阵(ASCⅡ字符)

每场显示17×10个汉字

或　　34×10个ASCⅡ字符

另外加一个状态行。

在低分辨彩色图形模式下

分辨率为40(横)×48(纵)

16种颜色

使用POKE指令访问地址－16301可令低分辨彩色图形与文

本混显。这时上部 40×40 作为低分辨图形显示区，下部四行作为文本显示区。

- 在高分辨彩色图形模式下

  分辨率为 280(横)×192(纵)

  6 种颜色

使用 POKE 指令访问－16301 可令图形文本混显。这时，上部 280×160 作为高分辨图形显示区，下部四行作为文本显示区。

·使用 GR 指令可置低分辨图形文本混显，使用 HGR 指令可置高分辨第一页图形文本混显；HGR2 指令置高分辨图形第二页

2. 颜色

在低分辨彩色图形下有 16 种颜色供选用，置色语句为

COLOR＝n

n 的取值与颜色的对应关系见表 10－1

表 10－1

n	颜色	n	颜色	n	颜色
0	黑	6	中蓝	12	浅绿
1	红	7	浅蓝	13	黄
2	深蓝	8	棕	14	绿蓝
3	紫	9	橙	15	白
4	深绿	10	灰 2		
5	灰 1	11	粉红		

在高分辨彩色图形下有八种颜色供选用，置色语句为

HCOLOR＝n

n 的取值与颜色的对应关系见表 10—2。

表 10－2

n	颜色	n	颜色
0	黑	4	黑 2
1	绿	5	红
2	蓝	6	黄
3	白 1	7	白 2

3. 其它

CEC-Ⅰ可运行中文程序，可外接盒式录音机、软盘驱动器、打印机、游戏操纵杆；可使用机器语言编程；可调用机器语言程序等等。

10.1.2 区分两种语言相同指令间的差异

CEC—BASIC、F BASIC 都是 BASIC 家族的成员，它们之间存在着密切的联系，但又各有特点。有些指令形式相同但功能各异，有些指令形式不同功能却相同。因此明确它们之间的联系和差别，尤为重要。为便于对照，特列表说明，见表 10－3。

表 10－3 两语言相同指令间的差异比较

内容	CEC—BASIC	F BASIC
变量：		
数值变量	取值＋/－9.99999999E＋37(浮点) －32768－＋32767(整型)	取值－32768～ ＋32767
串型变量	0－255 个字符	0--31 个字符
下标变量	三维或更多	最多二维
数组：	下标小于 10 可直接使用	必须先定义后使用
INPUT A$	可输入 255 个字符	只能输入 31 个字符
COLOR	COLOR＝n	COLOR X,Y,n
RND(X)	0－0.99999999 的数	0－(X－1)的数
PLAY	装入并执行磁带游戏	演奏音乐
NOT(0)	1	－1
NOT(1)	0	－2
PRINT X	紧凑打印	正数空一格打印
(x＝a)：		
当 x＝a 时	1	－1
当 x≠a 时	0	0
(x<>a)：		
当 x＝a 时	0	0
当 x≠a 时	1	－1

10.1.3 不同指令功能的对应实现方法

对于 F BASIC 中不具备的 CEC－BASIC 中的指令，要根据其指令功能采用相应指令实现，为便于对照，特列表于表 10－4。

表 10－4 不同指令功能的对应实现方法

CEC—BASIC 指令	CEC－BASIC 指令的功能	F BASIC 指令
TRACE	开跟踪	TRON
NOTRACE	关跟踪	TROFF
TAB	定位打印	POS(0) CSRLIN
SPC	空位打印	LOCATE
HOME	清屏(文本)	CLS
GR,HGR	图形区清屏	CLS,SPRITE OFF
GET A$	从键盘上读取一个字符	A$=INKEY$(0)
PLOT X,Y	在点(X,Y)显色点	LOCATE X,Y:PRINT
VTAB Y,HTAB X	选定打印的行列座标	LOCATE X,Y
HLIN X1,X2 AT Y	从点(X1,Y) 到点(X2,Y) 画一条水平线	FOR I=X1 TO X2: LOCATE I,Y: PRINT"－";:NEXT
VLIN Y1,Y2 AT X	从点(X,Y1)到点 (X,Y2) 画一条垂直线	FOR J=Y1 TO Y2: LOCATE X,J: PRINT"|";:NEXT
SCRN(X,Y)	测试在(X,Y)这一点的颜色	SCR$(X,Y Wn)
HPLOT X,Y	在(X,Y) 画点 X:0－279,Y:0－191 或 X1:0－279,Y:0－159	DEF SPRITE 或 DEF MOVE
ROT=X	图形旋转	DEF SPRITE
MUSIC X,Y	音乐语句,X:频率,Y:时间	PLAY
INT(A/B)	对 A/B 的商取整	A/B
C=A,A=B,B=C	交换 A、B 变量的内容	SWAP A,B
PEEK(-16336)	发出蜂鸣声	BEEP
PDL(X)	读游戏棒	STICK STRIG

10.1.4 区分两种语句结构的差异

要注意区分两种语句结构的基本差异，以便做到程序移植过程中胸中有数，现列表于表 10－5。

表 10－5

内　容	CEC—BASIC	F BASIC
1. 显示座标的差异		
文本方式：	每行 40 个字符	每行 28 个字符
低分辨图形	每行 40 个象素	每行 28 个象素
高分辨图形	每行 280 点	每行 255 点
2. 循环语句的差异		
(1)	NEXT 后可跟变量	NEXT 后不能跟变量
(2)	可执行小数步长	不能执行小数步长
(3)	从循环体转出，剩余的循环量不进栈保护	剩余的循环量要进栈保护
(4)	从内循环转到外循环的 NEXT，则执行外循环	从内循环转到外循环的 NEXT，仍执行内循环。
3. 清屏语句的差异	HOME	CLS
	GR	CLS
	HGR、HGR2	SPRITE OFF
4. 运算精度的差异		
运算结果	有小数	没有小数
5. 使用 STR $ 函数将数值转为字符时	不增加字符个数	要增加一个空格

10.1.5 提前作出是否可以移植的判断

根据两种语言的特点，提前作出是否可以移植的判断，以免增加无效劳动。

一般情况下，含有下述内容之一的程序不能移植：

1. 含有 SIN、COS、TAN、ATN、EXP、LOG、SQR 等函数的程序。

2. 过多地调用机器语言的程序。

3. 循环语句中有小数步长的程序

4. 进行多位小数计算的程序。

5. 运算数据或结果超出+32767～-32768 范围的程序。

6. 使用向量作图法，进行图形放大或缩小的程序。

7. 过多地使用 HPLOT 语句绘图的程序。

8. 其它情况，使用 F BASIC 难以模拟实现的程序.

以上所列的不能移植情况并非绝对。例如循环步长为小数的情况，在循环量较小，运算数据较小的情况下，也可以移植。如有下述原始程序：

```
10 FOR I=1 TO 2 STEP 0.2
20 A=A+I*I
30 PRINT A
40 NEXT I
```

我们可采用先扩大计算、后缩小输出的方法移植，移植的目标程序如下：

```
10 FOR I=10 TO 20 STEP 2
20 A=A+I*I
30 PRINT A/100".”A MOD 100
40 NEXT
```

这样输出的小数结果，稍欠紧凑，可通过字符函数进行改善。移植程序如下：

```
10 FOR I=10 TO 20 STEP 2
20 A=A+I*I:A$=STR$(A):AA=LEN(A$)
30 PRINT LEFT$(A$,AA-2)"." RIGHT$(A$,2)
40 NEXT
```

读者可试运行两个移植程序，比较结果，第二个程序较如实地模拟了原始程序的功能。

因此，在确定能否移植时，要具体情况具体对待，灵活掌握。

10.2 看懂原始程序 预计运行结果

确定了移植程序后，首先要根据原始程序语言的特点仔细阅读、分析程序结构、编程思路、计算方法和各语句的作用；然后模拟(追踪)程序运行，预计程序的运行结果；最后，把程序运行的最初结果、中间结果、最终结果记录或绘出备用。

例如程序 No. 9－24 八皇后题解过程的动态模拟程序，其原始程序是用 CEC－BASIC 编写的。程序清单见 No. 10—1。

```
1 REM No. 10--1
5 DIM C(8),T(8),S(15),(15):D=1
10 HOME:HGR2:HCOLOR=3
20 FOR I=0 TO 160 STEP 20:HPLOT I+60,0 TO I+60,160:
   HPLOT 60,I TO 220,I:NEXT
30 K=1
40 IF T(K)=1 THEN 80
50 IF S(8+D-K)=1 THEN 80
60 IF P(K+D-1)=1 THEN 80
70 C(D)=K:T(K)=1:S(8+D-K)=1:P(K+D-1)=1:CO=3:GO-
   SUB 600:D=D+1:IF D>8 THEN TA=TA+1:GET A$:GOTO
   90
75 GOTO 30
80 K=K+1:IF K<=8 THEN 40
90 D=D-1: IF D<1 THEN TEXT:PRINT "TOTAL=";TA:
   PRINT "***END***":END
100 K=C(D):CO=0:GOSUB 600:T(K)=0:S(8+D-K)=0:P(K+
    D-1)=0:GOTO 80
600 X=64+(K-1)*20:Y=12+(D-1)*20
610 HCOLOR=CO
620 HPLOT X,Y TO X+12,Y TO X+12,Y+5 TO X,Y+5 TO X,Y
```

```
    TO X-2,Y-8 TO X+2,Y-4 TO X+6,Y-10 TO X+10,Y-4
    TO X+14,Y-8 TO X+12,Y
640 RETURN
```

分析程序 No. 10—1 的绘图过程如下：

1. 通过 20 行画出的 8×8 的方格图。

2. 通过 70 行调用 600 行在方格中画出皇冠的图形（见图 10—1）。

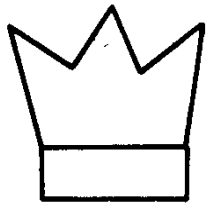
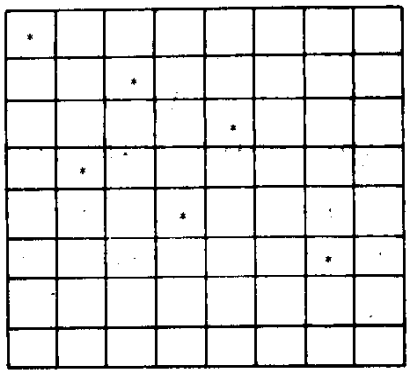


图 10－1

3. 通过 100 行调用 600 行擦去欲移动的皇冠图形。

因此，整个程序的运行过程就是：八个皇冠不停地改变显示位置，直到找出一种符合要求的摆放格式才停上移动。运行开始的图形如图 10－2。



注：图中“*”均为图10－1中的皇冠图形

图 10－2

再如金叶古塔搬移的动态模拟程序 No. 9－29 的原始程序清

单见 No. 10－2。

```
5  REM  No. 10－2
10  INPUT  "塔盘个数:";N
20  INPUT  "塔盘高度:";H
30  X(1)=40:X(2)=140:X(3)=239:Y=140:P(1)=N:A(1,1)=
    48:B=1:CALL  5576:HPLOT  0,141－H  TO  279,141－H
35  VTAB  10:PRINT  TAB(6)"A"TAB(18)"B"TAB(30)"C";
40  FOR  I=2  TO  N:A(1,I)=A(1,I－1)－2:NEXT
60  FOR  I=1  TO  N:P=I:GOSUB  290:NEXT
70  DIM  N(20),B(20),F(20),S(20)
80  I=0:N(1)=N:J=1
90  B(1)=1:F(1)=3:S(1)=2
100  GOSUB  120
110  END
120  I=I+1
130  IF  N(I)=0  THEN  I=I－1:RETURN
140  N(I+1)=N(I)－1
150  B(I+1)=B(I)
160  F(I+1)=S(I)
170  S(I+1)=F(I)
180  GOSUB  120
190  VTAB  20:HTAB  16:PRINT  J:J=J+1:B=B(I):F=F(I):
    PRINT  CHR$(7):GET  A$
200  VTAB  3:HTAB14:PRINT  CHR$(64+B)"→"CHR$(64+
    F):HCOLOR=4:P=P(B):GOSUB  300:P(F)=P(F)+1
210  A(F,P(F))=A(B,P(B)):P(B)=P(B)－1:B=F:  P=P(B):
    GOSUB  290
220  N(I+1)=N(I)－1
230  B(I+1)=S(I)
240  F(I+1)=F(I)
250  S(I+1)=B(I)
```

```
260  GOSUB  120
270  I=I-1
280  RETURN
290  HCOLOR=3
300  HPLOT  X(B)-A(B,P),Y-H*P  TO  X(B)-A(B,P),Y
     -H*(P+1)  TO  X(B)+A(B,P),Y-H*(P+1)  TO  X
     (B)+A(B,P),Y-H*P
320  RETURN
```

程序中由 30 行在屏幕中间画一条横线；由 35 行在横线下打印 A、B、C 三个塔座的标志；由 60 行调 290 行开始的子程序画出初始塔盘；由 200 行用“黑 2”调 300 行开始的子程序擦除欲移动的塔盘；由 210 行用“白 1”在新的位置上画出新近移动的塔盘。

整个程序的执行过程就是：执行到 70 行时，画出如图 10－3 的初始画面；以后通过 200、210 行模拟塔盘的移动过程，最终将塔盘全部移到 C 号塔座上，停机，见图 10－4。

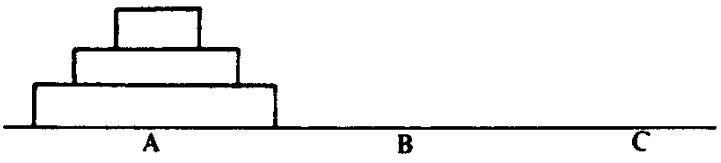


图 10－3　N＝3 的初始示意图

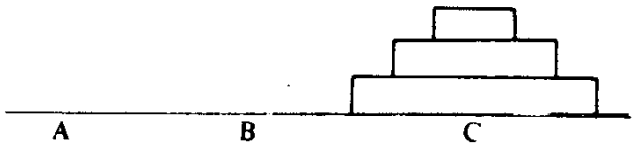


图 10－4　N＝3 结束示意图

10.3 分析程序特点 构思移植思路

通过对原始程序运行结果的分析，我们对移植程序的功能、作用、目的已有了明确的概念。后面要做的工作就是从 F BASIC 的特点出发，使用 F BASIC 语言实现原始程序的功能。也就是相同于原始程序功能的 F BASIC 语言程序的再编制、再创作。这一工作步骤，没有固定的模式，只有具体情况具体对待，采用具体的处理方法。

八皇后问题，原始程序中使用了高分辨作图指令画出皇冠；擦除皇冠使用了用黑色再画一遍的方法。在 F BASIC 中，我们可以考虑用显示卡通代替绘图，在欲画皇冠的座标上显示 SP，如使用丽莎小姐的各种姿式做皇后，将会比原始程序更形象、更逼真。同时，采用显示卡通代替绘图还可省去原始程序中擦除皇冠的操作，使程序更简洁易懂。8×8 方格的绘制有两种方法：一种是编程绘出，另一种是键控绘图，可灵活选用。移植的关键问题是显示卡通显示座标的计算和解决好显示卡通与方格的重合。这一工作可通过运行程序，试验调整。

对于金叶古塔搬移问题，已不能采取八皇后的同样方法用 SP 代替绘图（由于各塔盘的宽度从底向上逐渐缩小），只能在字符页面，使用 PRINT 语句打印某个字符或图案画塔盘。控制每行打印图案的数量来调整塔盘的宽度，以区分各号塔盘。塔盘的擦除，可采取原始程序的方法，用空格在确定的位置画一次塔盘来实现。可见，移植的关键是显示塔盘、擦除塔盘座标的确定以及显示、擦除塔盘的控制。因此，移植中，该部分的修正工作量较大。首先要调整 30 行中 X(1)、X(2)、X(3)、Y、A(1,1)的赋值，使其适应 F BASIC 文本显示方式下的做图要求；其次要改造 300 行的绘图子程序，使用 F BASIC 的有关语句进行模拟，最后还要做好塔盘绘制、

擦除的计算工作。

通过以上的规划、构思，对程序的移植有了明确的思路。选准了移植工作的主攻方向，从而有的放矢，做好了重点突破的准备工作。

另外，在分析原始程序的同时，还要注意修改原始程序中的错误。在发表公布的程序中，由于抄写的笔误，排版的失误，校验的疏忽，致使原始程序中可能存在着这样、那样的错误或遗漏，造成原始程序就不能正常运行或不能计算出正确的结果。显然这些错误或遗漏势必给移植工作增加困难。因此，必须消除错误于移植之前。如金叶古塔搬移的原始程序 No 10—2 中 30 行的 X(1)、X(3)的赋值就有错误。该错误体现在 300 行不能绘图。原因是 300 行绘图子程序的横向座标变化范围为 X(B)－A(B,P)～X(B)＋A(B,P)，当 B＝1、P＝1 时，有

X(1)－A(1,1)＝40－48＝－8

座标出现负值，显然不能够绘图，因此应把 X(1)的赋值由 40 改为 48，使绘图的初始座标最小为 0；当 B＝3、A(3,1)＝48 时有

X(3)＋A3,1)＝239＋48＝287

超过了屏幕的右边界，故破坏了画面，因此应把 X(3)的赋值由 239 改为 230，使绘图的终座标最大为 278。

10.4 移植目标程序 运行检验修正

程序的移植阶段是关键的一环，必须集中精力做好。在移植过程中，一般要遵循以下四条原则；

- 根据原程序的算法思路移植，一般不要作大的变动。
- 凡两语言中格式、功能相同的语句，可直接搬移。
- 对语句格式相同、功能不同的语句，要按照 F BASIC 的语法规则，调整出原始程序的功能。

·对于原始程序中，F BASIC 没有的语句要根据移植思路慎重改动。然后在程序运行中反复修改，以臻完善。

根据以上四条原则，八皇后问题的程序移植过程如下：

1. 根据移植思路，用显示卡通替代原程序中的皇冠，所以首先要先定义 8 个显示卡通。程序 No. 9—24 中的 30～50 行就是为定义显示 SP 而设置的。

2. 为压缩程序量，8×8 的棋盘绘制采用了键控绘图的方法，故仅在 20 行添一“VIEW”语句就完成了 No. 10—1 中 20 行的任务。

3. No. 9—24 中的 60～130 行对应着 No. 10—1 中的 30～100 行，其算法思路、程序格式基本上遵照原程序原样复制，仅省去了 100 行的

```
CO=0:GOSUB 600
```

两语句，该两语句的作用是用黑色画皇冠。（即擦除）由于显示 SP 改换位置后，原位置的显示自动消失，故不用专门擦除。

4. 移植中改动较大的就是绘图子程序 600～620 行。移植中是同下述语句实现的：

```
X=40+(K-1)*24
Y=60+(D-1)*16
SPRITE D-1,X,Y
```

移植的关键技巧在于根据 K、D 的取值确定某号 SP 的显示座标。读者可对照两程序，仔细领会其中的奥妙。

5. 程序移植完毕，运行检查，修改移植中的错误。在程序能够正常运行后，再仔细调整 SP 的显示位置，如 No. 9--24 中 140 行的 X、Y 值的计算常量 40、60，就是多次调整后确定的。

6. 运行程序，检验各运行结果是否全部正确。确认无误后则程序移植工作结束，可以投入使用。

古塔搬移程序的移植过程如下：

1. 在 No. 10—2 中有 P、X、A 三个数组没有定义，故首先要对数组补充定义。定义数组要确定数组的最大下标。由于 P、X 是指示塔座的数组，可确定 P、X 数组的最大下标为 3；由 40 行知两维数组 A 的第二下标最大为 N(即用户输入的塔盘个数；由 300 行知 A 数组的第一下标最大取值也为 3；因此 No. 9－29 中的 10 行补充定义了 P(3)、X(3)、A(3,N)三个数组。

2. 确定塔盘绘制程序的结构

No. 10—2 中 300 行使用 HPLOT 语句绘制塔盘，其横向座标变化范围为 0～279，即为

X(B)－A(B,P)～X(B)＋A(B,P)

根据移植思路使用 PRINT 语句绘塔盘，故考虑使用循环语句模拟。模拟程序如下：

```
A$="*"或 A$=CHR$(253)
FOR W=X(B)=A(B,P) TO X(B)+A(B,P)
LOCATE W,Y－P
RPINT A$;
NEXT
```

各塔盘的打印座标仍采纳了原程序的计算方法。

3. 根据塔盘绘制程序决定 X(1)、X(2)、X(3)、Y、A(1,1)的赋值。

在 F BASIC 中低分辨绘图的横座标变化范围为 0～27，那么，每个塔座上塔盘的最大宽度仅有 28/3＝9。我们选定 A 塔的起始座标 XA＝1；B 塔的起始座标 XB＝10；C 塔盘的起始座标 XC＝19；则每个塔盘的绘制范围为(即 X 方向的变化范围)：

A 塔，XA：　1～9

B 塔，XB：　10～18

C 塔，XC：　19～27

由 W 的初值

W＝X(B)－A(B,P)可得出方程组(1)

$$\begin{cases} XA-A(B,P)=1 \\ XB-A(B,P)=10 \\ XC-A(B,P)=19 \end{cases} \quad (1)$$

由 W 的终值

W=X(B)+A(B,P)可得出方程组(2)

$$\begin{cases} XA+A(B,P)=9 \\ XB+A(B,P)=18 \\ XC+A(B,P)=27 \end{cases} \quad (2)$$

解(1)、(2)方程组，可求出

$$\begin{cases} XA=5 \\ XB=14 \\ XC=23 \\ A(B,P)=4 \quad 即 \quad A(1,1)=4 \end{cases}$$

至此，我们计算确定了 X(1)、X(2)、X(3)。和 A(1,1)的赋值。但由 40 行

A(1,2)=A(1,1)−2=2

A(1,3)=A(1,2)−2=0

A(1,4)=A(1,3)−2=−2

……

可知若 A(1,1)=4 最多只能画两个塔盘，若把它们之间的差改为 1，即

A(1,2)=A(1,1)−1=3

最多只能画四个塔盘。显然使用范围太窄，因此，需要对 A(1,1)的赋值进行调整。程序 No. 9−29 中我们取 A(1,1)=19，比原值增加了 15，故 X(1)、X(2)、X(3)也都要相应增加 15，以保证起始座标不变。从而它们的赋值变为：

X(1)=20

X(2)=29

X(3)=38

A(1,1)=19

上述调整后又使 W 的循环终值出现了问题，可以算出三个塔盘的变化范围为

A 塔： 1～39

B 塔： 10～48

C 塔： 19～57

可见初值满足要求，而终值超出了屏幕横座标最大取值 27 的 30 点，故应令 W 的循环终值减去 30，方可保证绘图正常进行。

关于纵向座标 Y 的赋值较容易确定。No. 9－29 中取 Y＝18。当然也可以取其它值。

4. 关于打印塔盘的横线和塔座标志 A、B、C 的定位较简单，读者可对照 No. 9－29 的 40、50 行自行分析。

5. 塔盘的擦除处理。

把原擦除塔盘程序的 200 行中的 HCOLOR＝4 改为 A $ ＝“”，再调用绘图子程序即可达到塔盘擦除的目的。

6. 程序 No. 10—2 中的 80～280 行的算法程序基本照搬移植在 No. 9－29 中的 90～180 行，不多说明。

总之，程序的移植是一项既简单又复杂的工作。说其简单是有些程序可直接移入使用；说其复杂是指有些貌似简单的程序，直接移植却不能正常运行（在下一节介绍的插入法排序程序就会出现这一问题）。因此，要根据原始程序的编程思路，算法特点，对照 F BASIC 语法结构，反复思索、仔细推敲，谋定而后行。移植后还要在运行中反复检查修改，使其达到原始程序的各项要求，再现原始程序的各项功能。

10.5 程序移植示例

以上四节我们从整体方面提纲挈领地讨论了程序移植的方法、步骤和注意事项。本节将通过移植示例从细节方面介绍移植过

程中各种具体情况的具体处理，并结合介绍几个 PC 机 BASIC 程序的移植过程。

10.5.1 学生成绩排序

原始程序见 No. 10—3

```
5 REM No. 10—3
10 INPUT M
15 DIM T(M),N(M),A(M)
20 FOR I=1 TO M
30 READ T(I)
40 READ N(I)
50 READ A(I)
60 NEXT I
70 FOR I=1 TO M-1
80 FOR J=I+1 TO M
90 IF T(I)>=T(J) THEN 130
100 LET T1=T(I)
102 LET N1=N(I)
106 LET A1=A(I)
110 LET T(I)=T(J)
112 LET N(I)=N(J)
116 LET A(I)=A(J)
120 LET T(J)=T1
122 LET N(J)=N1
126 LET A(J)=N1
130 NEXT J
140 NEXT I
150 FOR I=1 TO M-1
160 FOR J=I+1 TO M
170 IF T(I)<>T(J) THEN 200
180 IF N(I)<N(J) THEN 200
```

```
190 LET N1 =N(I)
191 LET A1=A(I)
193 LET N(I)=N(J)
195 LET A(I)=A(J)
196 LET N(J)=N1
198 LET A(J) =A1
200 NEXT J
210 NEXT I
220 FOR I=1 TO M-1
230 FOR J=I+1 TO M
240 IF T(I)<> T(J) THEN 300
250 IF A(I)<=A(J) THEN 300
260 LET A1 =A(I)
262 LET N1 =N(I)
270 LET A(I)=A(J)
272 LET N(I)=N(J)
280 LET A(J) A1
282 LET N(J)=N1
300 NEXT J
310 NEXT I
320 FOR I=1 TO M
330 LPRINT I;T(I);N(I);A(I)
350 NEXT I
400 DATA 90,1010,19
410 DATA 90,1011,20
.....
```

该程序平铺直叙，结构简单。程序中 70～140 行对成绩进行降序排列；150～210 行对学号进行升序排列；220～310 行对学生年令进行升序排列；320～350 行输出排序结果。程序中的 100～126 行 9 行程序仅为交换 T(I)与 T(J)、A(I)与 A(J)、N(I)与 N(J)三

对变量之间的内容，故在移植中我们可以用

```
SWAP T(I),T(J):SWAP A(I),A(J):SWAP N(I),N(J)
```

一行程序代替。同样，190～198 行、260～282 行都可如此处理。

该程序没有特殊结构和特殊计算，故移植过程极为简单，读者可尝试一下。移植完成的程序见 No. 8－48。

10. 5. 2 插入法排序

原始程序见 No. 10—4。

```
1 REM No. 10－4
5 PR#3:HGR2:HOME
10 VTAB 5:PRINT "插入法排序"
15 INPUT "排序数目 N=";N:INPUT "区间最小值 A=";A:INPUT
   "区间最大值 B=";B
20 DIM A(N)
25 FOR I=1 TO N : A(I)=A+INT(B * RND(1)) : PRINT A(I);"
   "; : NEXT : PRINT
30 GET A$
35 FOR I=2 TO N
40 A=I－1
45 FOR J=A TO 1 STEP －1
50 IF A(I)>A(J) THEN 60
55 K=J:GOTO 70
60 NEXT
65 K=0 :B=A(I)
70 IF A=K THEN 95
75 FOR J=A TO K+1 STEP －1
80 A(J+1)=A(J)
85 NEXT
90 A(K+1)=B
95 NEXT
100 PRINT:PRINT CHR$(7):GET A$
```

```
105 FOR I=1 TO N:PRINT A(I);"";:NEXT
110 PRINT:PRINT "结束!":END
```

程序中 10～25 行为人机对话，由用户输入排序的数量、数据的区间值，然后由 25 行随机产生 N 个待排序的数字。

35～95 行为排序程序。其中 55 行有遗漏致使原程序不能正常运行，故应补充完整。55 行的完整程序如下：

```
55 K=J: B=A(I): GOTO 70
```

100～110 行是排序结果输出和结束处理程序。

程序 No. 10－4 没有特殊的语法结构和绘图程序，只要删去中文内容或用英语、汉语拼音表述，应该很容易移植到 F BASIC 中。移植程序见 No. 10－5。

```
5 REM No. 10－5
10 CLS
15 INPUT "N=";N:INPUT "min. =";A:INPUT "max. =";B
20 DIM A(N)
25 FOR I=1 TO N:A(I)=A+RND(B－A) : PRINT A(I); : NEXT
   : PEINT
35 FOR I=2 TO N
40 A=I－1
45 FOR J=A TO 1 STEP－1
50 IF A(I)>A(J) THEN 60
55 K=J:B=A(I):GOTO 70
60 NEXT
65 K=0:B=A(I)
70 IF A=K THEN 95
75 FOR J=A TO K+1 STEP－1
80 A(J+1)=A(J)
85 NEXT
90 A(K+1)=B
95 NEXT
```

```
105 FOR I=1 TO N : PRINT A(I); : NEXT
110 END
```

运行程序，将会发现 No. 10－5 根本不能排序，仔细分析排序程序 35～95 行也没有错误。问题出在哪里呢？加入 36 行

```
36 PRINT:PRINT"I="I
```

检查循环变量 I 的运行情况。多次运行发现，I 仅循环 3、5 次就结束了排序过程，根本循环不到 N 次。这就是不能排序的根本原因。

分析产生这一现象的原因，主要是 F BASIC 中的 NEXT 后面没有变量，故当程序执行到 55 行转到 70 行后，再通过 70 行转到 95 行的 NEXT。这一个 NEXT 本应是循环变量 I 的，但由于 NEXT 后没有变量，系统仍把它认为是 J 的 NEXT，从而造成 I 失去返回语句不能循环到终值。改进的方法是使用 GOTO 语句代替内循环语句。具体方法是改编 No. 10－5 中的 45 行和 60 行。

```
45 J=A
60 J=J-1:IF J>0 THEN 50
```

改进后的插入法排序程序见 No. 8－47。现在可以正常排序了。

小结：

在程序的移植过程中，对多重循环嵌套的情况，若内循环体中存在转向外循环的语句时，一般要用 GOTO 语句代替该循环，以避免运行中出现错误或遗漏结果。

10.5.3 奇偶数字方阵

边线等和奇偶数字方阵的原始程序见 No. 10－6。

```
5 REM No. 10-6
10 CLS,这是一个求取 N 阶幻方的实用程序
40 PRINT:INPUT "请输入阶数 N[N>2]";N
50 PRINT:I=1:P=1:G=N AND 3
55 IF N<3 THEN PEINT "阶数小于 3。重新输入。":RUN 40
60 L=N/(1+ABS(G=2)):J=INT(L/2+0.5):IF G=0 THEN J=1
```

```
70 DIM S(L,L):FOR T=1 TO L*L:R=INT(T/L)
80 S(I,J)=T:IF G>0 THEN 150
90 IF T/L=R THEN I=I+1:J=(1+L)*SGN(I AND 2)
100 P=1-P:IF P=0 THEN I=L-I+1
110 J=J+1-(R+1 AND 2):GOTO 180
150 IF T/L=R THEN I=I+1 : GOTO 180
160 I=I-1+ABS(I=1)*L
170 J=J+1-ABS(J=L)*L
180 NEXT : T=T-1:P=LEN(STR¥(T)):K=P*N/2-8
185 K=ABS(N>6)*K+1:FOR I=1 TO P:Y¥=Y¥+CHR¥(35)
190 NEXT:K=K+ABS(N>L)*N/2:PRINT TAB(K);
195 PRINT "下面是";N;"阶幻方":IF N>L THEN 300
200 FOR I=1 TO N : FOR J=1 TO L
210 PRINT USING Y ¥;S(I,J);:NEXT:PRINT:NEXT
220 PRINT TAB(K);:其特征值为:;(N*N+1)/2*N
230 PRINT :INPUT "还想打印请敲 Y 键";Y
240 IF Y¥="Y"OR Y¥="y"THEN RUN 40 ELSE END
300 R=INT(L/2)+1:FOR S=1 TO N:I=S-ABS(S>L)*L
310 FOR P=1 TO N:J=P-ABS(P>L)*L:G=(J>1 OR I<>R)
320 G=3*T*(ABS(S>L)<>G*(J<R)+(I=R)*(J=R))
330 IF P>L THEN G=T+ABS((S>L)=(1+J<R))*T
340 PRINT USING Y$+"#";S(I,J)+ABS(G);:
350 NEXT: PRINT:NEXT:GOTO 220
```

该程序是在 PC 机上运行的程序。程序中由于"$"被打印成¥而出现了我们陌生的变量名和函数名，这种情况是由于日本打印机为处理方便，将美元标志改成日元标志所造成。"¥"不是标准 ASCII 字符，键盘上一般找不到，实际输入程序时原是键入"$"符的。

还有两个陌生语句 PRINT USING：为按设定的格式打印数据的语句，语句后可以跟格式说明符，不同的格式说明符代表不同

的意义。如 340 行中的

PRINT USING Y$+"#";

"#"代表一个数位，可使整数部分按右对齐的方式显示。

·IF－THEN －ELSE 是如果……则……否则的意思，作用是，如果条件成立则执行 THEN 后面的语句；如果条件不成立则执行 ELES 后面的语句。

很显然，程序 No. 10－6 比以上几个程序的程植工作量要大得多。程植中，可对该程序的有关语句作如下的移植处理：

1. 对于运算符语句由于运算式成立其植恒为"－"；不成立其值恒为"0"，故可去掉对其取绝对值的操作。如

原 60 行中的　　L=N/(1+ABS(G=2))

改为　　L=N/(1－(G=2)

再如原 160 行　　I=I－1+ABS(I=1)*L

移植为 I=I－1－(I=1)*L

2. 对于 60 行中

J=INT(L/2+0.5)的语句，在移植中改为　J=(L+1)/2

3. 对于 90 行、150 行中的 IF T/L=R THEN……

其原意是指：如果 T 能够被 L 整除且其商等于 R 才执行 THEN 后面的语句。由于 F BASIC 有对商自动取整的特点，若直接移植，则不能确定是否整除。因此，在移植中必须做如下处理

IF T MOD L=0 AND T/L=R THEN……

4. 原程序中的 185～190 行的

180 P=LEN(STR$(T)):K=P*N/2－8

185 K=ABS(N>6)*K+1:FOR I=1 TO P:Y$=Y$+CHR$(35)

190 NEXT:K=K+ABS(N>L)*N/2:PRINT TAB(K);

是为输出的魔方整齐美观作准备的程序，移植中可删去。故 180 行改为

180 NEXT :T=T－1

195 行改为

```
195 IF N>L THEN 300
```

5. 对于 210 行中的 PRINT USING Y$;S(I,J),……可移植为

```
210 PRINT S(I,J);:NEXT:PRINT:NEXT
```

6. 对于 220 行中求特征值的计算公式

```
(N*N+1)*N/2
```

要改为先乘 N 再除以 2 的形式，以保证计算特征值的精确性。即为

```
(N*N+1)*N/2
```

7. 对于 240 行中 ELSE 语句则移植为

```
240 IF Y$="Y" THEN RUN 40
250 END
```

8. 对于 340 行中的 PRINT USING Y$+"#";S(I,J)+ABS(G);:则移植为

```
340 PRINT S(I,J)+ABS(G);
```

9. 对 320 行中的"(J<R)"由于没法添"-"号，所以要加 ABS 函数，移植为

```
320 G=3*T*(ABS(S>L)<>G*ABS(J<R)+(I=R)*(J=R))
```

除以上 9 点外，其它部分基本参照原始程序移植。移植目标程序见 No. 9—22。

10.5.4 带农历的月历程序

在第九章，我们曾讨论编写了万年月历程序 No. 9—43。该程序固然简洁易懂，打印的月历清晰直观、方便实用，但美中不足的是没有农历。现在，结合程序的移植介绍带农历的月历程序的编写。

原始程序在 PC 机上调试成功，发表在《电子与电脑》杂志 1992 年 1 月号的期刊上，现抄录如下(见程序 No. 10—7)：

```
5 REM No. 10—7
```

```
10 SCREEN 0:WIDTH 80:COLOR 1,6:CLS
20 YN=1991
30 TN=2:NL=16:DIM D(12)
40 FOR K=1 TO 7:READ NAM$(K):NEXT K
50 FOR K=1 TO 12:READ D(K):NEXT K
60 PRINT TAB(32);"* * *";:COLOR 4:PRINT YN;:COLOR 1:
   PRINT "* * *":PRINT
70 READ X:NX=1
80 FOR I=1 TO 12
90 IF ((I-1)/3)=INT((I-1)/3)THEN GOSUB 390
100 S=1
110 FOR K=1 TO D(I)
120 IF TN=7 THEN COLOR 13
130 PRINT TAB(TN*10);USING "\\";STR$(K);
140 IF NL<>1 THEN COLOR 3:GOTO 190
150 COLOR 5
160 IF NX=2 THEN PRINT USING "-(Y##-)";12;:GOTO 180
170 PRINT USING "-(-Y##-)";NX-2
180 COLOR1:GOTO 210
190 IF TN=7 THEN COLOR 13
200 PRINT USING "-(##-)";:COLOR 1
210 IF NL<>X THEN NL=NL+1:GOTO 230
220 READ X:NL=1:NX=NX+1
230 IF TN<>7 THEN TN=TN+1:GOTO 270
240 TN=1:S=S+1:PRINT
250 IF S<>3 THEN 270
260 COLOR 4:PRINT "*";I;"*";:COLOR 1
270 NEXT K
280 IF TN=1 THEN 300
290 PRINT
300 IF I/3 <>INT (I/3) THEN PRINT:GOTO 340
```

```
310 GOSUB 390:PRINT:PRINT
320 A$=INKEY$:IF A$="" THEN 320
330 IF A$<>CHR$(13) THEN BEEP:GOTO 320
340 NEXT I
350 END
360 DATA M1,T2,W3,T4,F5,S6,SU
370 DATA 31,28,31,30,31,30,31,31,30,31,30,31
380 DATA 30,30,29,30,29,29,30,29,29,30,29,30,30
390 COLOR 14:FOR J=1 TO 6
400 PRINT TAB(3+J*10);NAM$(J);
410 NEXT J
420 COLOR 4:PRINT TAB(73); NAM$(7)
430 COLOR 1
440 RETURN
```

1. 程序 No. 10—7 的功能

(1)每季度首未都打印星期标志。其中周一到周六为黄色，星期天为红色。

(2)周一至周六的公历用蓝色打印，紧随其后的农历(除每月的第一天外)都用青色打印。

(3)星期日的公历和农历都用亮洋红打印(每月第一天除外)。

(4)农历每月的第一天都用洋红打印，并且用“Y＋该农历月份数”表示，以便识别。

(5)在每月的左侧打印出该公历月的名称(用数字表示)。

2. 程序说明：

①变量说明：

YN——公元年(No. 10—7 中为 1991 年)

TN——星期(初值表示元旦是星期几)

NL——农历的日期(初值表示元旦是初几)

D 数组——公历每月的天数

NAM $ 数组——星期标志。

X——农历每月的天数

NX——农历的月份数

S——每月打印行数指针。

2. 语句说明：

(1)10～70 行，初始赋值；把星期标志装入 NAM $ 数组；把公历每月的天数装进 D 数组；打印年号；读入公历一月份时农历月的天数。

(2)120～130 行打印公历的日期

(3)160～170 行是专为处理公历、农历的月份不一致而设置的，每年的情况各不相同。

(4)190～200 行打印农历的日期。原程序的 200 行有遗漏，正确的语句应为：

PRINT USING "－(##—)";NL :COLOR 1

(4)100 和 250 两行语句是为了让公历月份名称打印在该月第三行的位置上。

(5)320～330 行为显示控制语句，每打印一个季度暂停，等待用户指示再打印下一个季度。可以不要。

(6)390～440 行为打印星期标志子程序。

(7)370～380 行分别存放公历和农历每月的天数。其中 380 行的前两个数据(30,30)为农历 1990 年的 11 月、12 月的天数。

(8)80～340 行为主体程序。其中 110～270 行由 K 循环控制每一个月份各天的打印。I 循环控制 12 个月的打印。

3. 使用说明：

使用 No. 10—7 时要根据打印年度改变 20 行、30 行中 YN、TN、NL 的初值；调整 160、170、380 三行的有关数值。如 1991 年元旦是星期二，农历是 11 月 16。故 YN＝1991；TN＝2；NL＝16；160 行中 NX＝2 打印 12；380 行中前两个数字为 90 年 11、12 两

月的天数。如果欲打印 1992 年的月历，可作如下改动：

1992 年的元旦是星期三，农历：十一月二十七日

1991 年农历的 11 月、12 月都是 30 天

1992 年农历各月的天数依次如下：

29,30,30,29,29,30,29,29,30,29,30,30

因此有

YN＝1992,TN＝3,NL＝27

380 DATA 30,30,29,30,30,29,29,30,29,29,30,29,30,30

同时，由于 1992 年是闰年，公历二月份为 29 天故 370 行中的 28 还应改为 29。

由于元旦是农历 11 月份，所以 160、170 两行不变。(若元旦是 12 月份，则要把 160 行中的“12”改为“1”，170 行中的“NX－2”改为“NX－1”)。

4. 陌生语句说明：

(1)WIDTH 设置屏幕显示宽度语句。

WIDTH 80：表示把屏幕置成横向显示 80 个字符的形式。

(2)PRINT USING “\(n 个空格)\”其功能是打印字符串的前(n＋2)个字符。

现在 130 行是“\\”，中间没有空格，则只打印字符串的前两个字符。

(3)COLOR n 置色语句。程序中用到的几种颜色与 n 的对应关系如下：

1：蓝　3：青　4：红　5：洋红　13：亮洋红　14：黄

下面，我们讨论程序 No. 10—7 的移植。

1. 预计运行结果

根据程序说明，追踪程序运行，No. 10—7 的第一次运行结果将如图 10—5 所示。按回车键将打印第二季度的月历。

```
                    * * * 1991 * * *

      M1    T2    W3    T4    F5    S6    SU
                  1 16  2 17  3 18  4 19  5 20  6 21
      7 22  8 23  9 24  10 25 11 26 12 27 13 28
*1*   14 29 15 30 16Y12 17 2  18 3  19 4  20 5
      21 6  22 7  23 8  24 9  25 10 26 11 27 12
      28 13 29 14 30 15 31 16

*2*(略)

                                  1 15  2 16  3 17
      4 18  5 19  6 20  7 21  8 22  9 23  10 24
*3*   11 25 12 26 13 27 14 28 15 29 16Y2 17 2
      18 3  19 4  20 5  21 6  22 7  23 8  24 9
      25 10 26 11 27 12 28 13 29 14 30 15 31 16
      M1    T2    W3    T4    F5    S6    SU
```

图 10—5　程序 No. 10—7 的运行结果

2. 构思移植思路

(1)为简化程序,删去染色语句。

(2)由于要同时打印公历、农历,故每天至少要占用四个字符位置,每周七天,共计占 28 个字符位置,正好占满 F BASIC 的一行。故只能采取紧凑格式打印。因此各日期必须化为字符串并整形后才能输出。

(3)由于两日期间没有空格,为便于区别,把每个星期的标志字符分开打印。如"M1"打印成"M 1",并使公历的低位数对齐"M",使农历的低位数对齐"1"。

(4)因日期刚好占满一行,已没有空位供打印月份,故把月份移到屏幕中间另起一行打印。

(5)因屏幕纵向只有 24 行,容不下一个季度的数据,故改为每次打印一个月后暂停。

(6)农历每月的第一天仍打印月份,但已没有空间打印 Y 标志,故省去 Y。

(7)由于每次只打印一个月的日历,故省去底行的星期标志。

③移植程序

根据移植思路,重点要解决两个问题:一个是打印定位问题。现在在 X 方向,要从 0 开始每隔 4 个位置打印一天的日期,故把原程序中 130 行的 TAB(TN * 10)改为 TN * 4,同时对 NT 的初值要减 1。第二个问题是要解决好输出项的字符化整形。对于公历 K 进行如下处理:

```
G $ =STR $ (K):IF LEN(G $ )>2 THEN G $ =RIGHT $ (G $ ,
LEN(G $ )-1)
```

对于农历 NL 也进行相似处理:

```
M $ =STR $ (NL):IF LEN(M $ )>2 THEN M $ =RIGHT $ (G $ ,
LEN(G $ )-1)
```

上述处理的目的是去除数字字符化后前面出现的空格。

移植的目标程序见 No. 10—8

```
5 REN No. 10—8
10 CLS:YN=1991:TN=2:NL=16
20 TN=TN-1:DIM D(12),N$(7)
30 FOR K=1 TO 7:READ N$(K):NEXT
40 FOR K=1 TO 12:READ D(K):NEXT
50 PRINT "                    * * * "YN"   * * * ":PRINT
60 READ X: NX=1
70 FOR I=1 TO 12:PRINT "              * * * * * "I" * * * * * ":
   PRINT
80 FOR J=1 TO 7 : ;PRINT ""N$(J);:NEXT:PRINT
100 FOR K=1 TO D(I)
110 G$=STR$(K):IF LEN(G$)>2 THEN G$=RIGHT$(G$,
    LEN(G$-1)
120 LOCATE TN*4,CSRLIN:PRINT G$;
130 IF NL<>1 THEN 160
140 IF NX=2 THEN M$="12":GOTO 180
150 M$=STR$(NX-2):GOTO 170
160 M$=STR$(NL)
170 IF LEN(M$)>2 THEN M$=RIGHT$(M$,LEN(M$)-1)
180 PRINT M$;
190 IF NL<>X THEN NL=NL+1:GOTO 210
200 READ X:NL=1:NX=NX+1
210 IF TN<6 THEN TN=TN+1:GOTO 230
220 TN=0:PRINT
230 NEXT
240 A$=INKEY$(0):PRINT:PRINT
250 NEXT
260 DATA M 1,T 2,W 3,T 4,F 5,S 6,S 7,
270 DATA 31,28,31,30,31,30,31,31,30,31,30,31
280 DATA 30,30,29,30,29,29,30,29,29,30,29,30,30
```

运行程序，将打印如图 10—6 的运行结果，按任一键后则打印下一月份的月历……。

```
          * * * 1991 * * *

          * * * * * 1 * * * * *

M 1     T 2     W 3     T 4     F 5     S 6     S 7
        1 16    2 17    3 18    4 19    5 20    6 21
7 22    8 23    9 24    10 25   11 26   12 27   13 28
14 29   15 30   16 12   17 2    18 3    19 4    20 5
21 6    22 7    23 8    24 9    25 10   26 11   27 12
28 13   29 14   30 15   31 16
```

图 10—6

读者可逐一月份打印检验，各月份都能正确地打印，程序的移植是成功的。但成功之余，我们总是感到不是十分满意。不满意之处就是一行一串数字，公历、农历不易分别。原始程序靠使用两种颜色打印公、农历以示区别，我们应如何区别呢？虽然可以用 COLOR 语句染色，但我们知道，COLOR 语句一次染 4 个方块，那么在同一颜色中必然公、农历混杂，将更不容易分辨。显然，使用 COLOR 语句行不通。还有没有其它方法呢？

查阅符号代码表 B，发现 10 进制代码 128～137 为汉字的“一至十”。这为我们的数字显示开辟了新天地。若能使用汉字的“一、二、三……”表示农历的日期，不就能达到公、农历易于分辨的目的了吗。问题是如何使用这十个中文数字。

如果我们定义一个字符串数组 C$(30)，把用中文数字表示的日期“一至三十”装进 C$ 数组中，那么在程序 No. 10—8 中以 NL 为 C$ 数组变量的下标，打印 C$(NL)即可达到目的。关键是

如何把汉字装入C$数组。

如果我们采取C$(1)=CHR$(128)、C$(2)=CHR$(129)、C$(3)=CHR$(130)……C$(30)=CHR$(130)+CHR$(137)的方法逐个赋值虽然可达到目的，但程序量太大，是下策。如果我们定义一个中间数组B$(9)，把中文数字"一～十"装入B$的对应下标变量中，再通过循环语句装入C$数组则为上策。

试验装入程序如下：

```
300 DIM B$(9),C$(30)
310 FOR J=1 TO 9:B$(I)=CHR$(127+I):NEXT
320 B$(0)=CHR$(137)
330 FOR I=1 TO 30
340 IF I<10 THEN C$(I)=" "+B$(I):GOTO 360
350 C$(I)=B$(I/10)+B$(I MOD 10)
360 PRINT C$(I)" ";:NEXT
```

运行程序则打印出

一　二　三　四　五　六　七　八　九

十　十一　十二　十三　十四　十五　十六　十七　十八

十九　二十　二一　二二　二三　二四　二五　二六　二七

二八　二九　三十

运行结果表明，试验程序是成功的。把它作为子程序移入程序No. 10—8中，适当地改变主程序的有关语句就可达到汉字显示农历的目的。为便于农历月份的区别，对农历月份仍保留原来的形式。改造后的程序见No. 10—9。

```
5 REM No. 10—9
10 CLS:YN=1991:TN=1:NL=16
20 DIM D(12),N$(7),B$(9),C$(30)
30 FOR K=1 TO 7:READ N$(K):NEXT
40 FOR K=1 TO 12:READ D(K):NEXT
```

```
50 PRINT "          * * * "YN" * * * ":PRINT
60 READ X:NX=1:GOSUB 290
70 FOR I=1 TO 12:PRINT "          * * * * * "I" * * * * * ":
   PRINT
80 FOR J=1 TO 7:PRINT" "N$(J)" ";:NEXT:PRINT
90 FOR K=1 TO D(I)
100 G$=STR$(K):IF LEN(G$)>2 THEN G$=RIGHT$(G
    $,LEN(G$)-1)
110 LOCATE TN*4,CSRLIN:PRINT G$;
120 IF NL<>1 THEN 170
130 IF NX=2 THEN M$="12":GOTO 160
140 M$=STR$(NX-2)
150 IF LEN(M$)>2 THEN M$=RIGHT$(M$,LEN(M$)-1)
160 PRINT M$;
170 PRINT C$(NL);
180 IF NL<>X THEN NL=NL+1:GOTO 200
190 READ X:NL=1:NX=NX+1
200 IF TN<6 THEN TN=TN+1:GOTO 220
210 TN=0:PRINT
220 NEXT
230 A$=INKEY$(0):PRINT:PRINT
240 NEXT
250 END
260 DATA M1,T2,W3,T4,F5,S6,S7
270 DATA 31,28,31,30,31,30,31,31,30,31,30,31
280 DATA 30,30,29,30,29,29,30,29,29,30,29,30,30
290 FOR J=1 TO 9:B$(J)=CHR$(127+J):NEXT
300 B$(0)=CHR$(137)
310 FOR J=1 TO 30
320 IF J<10 THEN C$(J)=" "+B$(J):GOTO 340
330 C$(J)=B$(J/10)+B$(J MOD 10)
```

```
340 NEXT
350 RETURN
```

运行程序将打印图 10—7 的结果。

现在公、农历分辨清晰，结果美观整齐，令人十分满意。达到了理想的移植效果。

```
                    * * * 1991 * * *
                  * * * * * 1 * * * * *
M1      T2      W3      T4      F5      S6      S7
        1 一六  2 一七  3 一八  4 一九  5 二十  6 二一
7 二二  8 二三  九二四  10 二五 11 二六 12 二七 13 二八
14 二九 15 三十 16 一   17 二   18 三   19 四   20 五
21 六   22 七   23 八   24 九   25 一十 26 一一 27 一二
28 一三 29 一四 30 一五 31 一六

                  * * * * * 2 * * * * *
M1      T2      W3      T4      F5      S6      S7
                                1 一七  2 一八  3 一九
4 二十  5 二一  6 二二  7 二三  8 二四  9 二五  10 二六
11 二七 12 二八 13 二九 14 三十 15 一   16 二   17 三
18 四   19 五   20 六   21 七   22 八   23 九   24 一十
25 一一 26 一二 27 一三 28 一四
```

图 10—7

小结：

①程序移植中，若要求以紧凑格式输出时，可采用数值化字符的方法进行整形输出。

②当程序中要求以不同的字符形式输出数字时，可以考虑使用中文数字。

③在移植中要敢于删削，不要拘泥原始程序的显示形式，要根据 F BASIC 的优点、优势大胆改革，只要保留原程序的中心内容和主要功能即可。

10.5.5 哥德巴赫猜想

著名数学家哥德巴赫(Goldbach,1690～1764)是德国人,1725年当选为俄国彼得堡科学院院士。哥德巴赫猜想是他在1742年与大数学家欧拉的通信中提出来的,其内容就是关于正整数和素数之间关系的两个推测。用通俗的语言描述就是:

1. 每一个不小于6的偶数都是两个奇素数之和。

2. 每一个不小于9的奇数都是三个奇素数之和。

这就是哥德巴赫猜想的内容,通常简称为偶数猜想和奇数猜想。

在数论中用符号{a,b}表示:任一个大偶数都可表示为一个不超过a个素因子的乘积和另一个不超过b个素因子的乘积之和。据有关数学文献介绍,1920年后,有人证明了{9,9}。1924年{7,7}被证明。1932年{6,6}被证明。1937年证明了{5,7}、{4,9}、{3,15}。1938年证明了{5,5}。1939年{4,4}被证明。1956年我国数学家王元证明了{3,4},1957年又证明了{2,3}。1962年潘承洞证明了{1,5}1962年王元和潘承洞又分别证明了{1,4}。直到1966年和1978年由陈景润证明了{1,2}。现在正一步步向{1,1}逼近。

下面,我们介绍用BASIC语言验证哥德巴赫两个猜想的程序。

①哥德巴赫偶数猜想

原始程序在LASER310和IBM PC/XT微机上通过。程序清单见No. 10—10。

```
5 REM No. 10—10
10 INPUT A
20 B=INT(A/2)
30 M=0
40 FOR Y=B TO A
50 X=A-Y
```

```
60 IF Y=2*INT (Y/2) THEN 250
70 IF X=2* INT (X/2) THEN 250
80 IF X=3 THEN 200
90 IF X=1 THEN 250
100 SY=INT (SQR(Y+0.5))
110 SX=INT (SQR(X+0.5))
120 IF SX=2* INT (SX/2) THEN 150
130 SS=SX+2
140 GOTO 160
150 SS=SX+1
160 FOR N=3 TO SX STEP 2
170 IF Y=N*INT(Y/N) THEN 250
180 IF X=N*INT(X/N) THEN 250
190 NEXT N
200 FOR N=SS TO SY STEP 2
210 IF Y=N*INT (Y/N) THEN 250
220 NEXT N
230 M=M+1
240 PRINT M,X;"  ";Y
250 NEXT Y
280 PRINT
290 PRINT A,M
300 GOTO 10
```

原始程序给出了输入 100 的运行结果如下：

```
1       47      53
2       41      59
3       29      71
4       17      83
5       11      89
6       3       97
100     6
```

运行结果表明：偶数 100 有 6 组(1+1)的素数对，用数论符号表示即为

D(100)=6

下面，我们讨论如何把 No. 10—10 移植到 F BASIC 中。

阅读 No. 10—10 发现，程序移植的难点在 100、110 两行中的“SQR”函数。SQR 函数是求平方根函数，F BASIC 中没有这一函数，如果采用第六章介绍的求幂子程序，将使程序极为庞大。仔细分析 No. 10—10 发现，程序中对平方根的要求并不精确，引进平方根函数的目的是合理地减小循环终值以减少循环次数，提高程序的执行速度。因此，我们可以用倒推的方法求出平方根。对于

SY=INT(SQR(Y+0.5))

我们可用如下子程序求出：

```
100 FOR I=2 TO Y
110 IF I*I>Y THEN SY=I:GOTO 130
120 NEXT
130 RETURN
```

为了避免 F BASIC 从循环体中转出的差错，我们可以用 GOTO 语句代替这一循环语句，形式如下：

```
100 I=2
110 IF I*I>Y THEN SY=I:RETURN
120 I=I+1:GOTO 110
```

解决了求平方根的问题就可移植程序了。目标程序见 No. 10—11。

```
5 REM No. 10—11
10 INPUT "Input  an  EVEN number";A:B=A/2
20 FOR Y=B TO A: X=A-Y
30 IF Y MOD 2=0 R X MOD 2=0 THEN 180
40 IF X=3 THEN 140
50 IF X=1 THEN 180
```

```
60 W=Y:GOSUB 210:SY=P
70 W=X:GOSUB 210:SX=P
80 IF SX MOD 2=0 THEN 100
90 SS=SX+2:GOTO 110
100 SS=SX+1
110 N=3
120 IF Y MOD N=0 OR X MOD N=0 THEN 180
130 N=N+2:IF N<=SX THEN 120
140 N=SS
150 IF Y MOD N=0 THEN 180
160 N=N+2:IF N<=SY THEN 150
170 M=M+1:PRINT M,A"="X"+"Y
180 NEXT
190 PRINT:PRINT"D("A")="M
200 CLEAR:GOTO 10
210 P=0:K=2
220 IF K*K>W THEN P=K:RETURN
230 K=K+1:GOTO 220
```

程序的移植中，仍遵照原始程序的计算方法，不同的是计算形式稍有变更：

(1)No. 10—10 中所有判断整除的语句 60、70、120、170、180、210 行均改为 MOD=0 的形式。

(2)所有内循环均使用 GOTO 语句代替。

(3)210～230 行为求平方根子程序，供 60、70 两行调用。

程序结构较简单，读者可自行对照分析。

运行程序，输入 20 则打印

```
 1          20=7+13
 2          20=3+17
D(20)=2
```

输入 100 则打印

1　　100=47+53

2　　100=41+59

3　　100=29+71

4　　100=17+83

5　　100=11+89

6　　100=3+97

D(100)=6

与原始程序运行结果相同。

运行程序输入 200,打印

1　　200=97+103

2　　200=73+127

3　　200=61+139

4　　200=43+157

5　　200=37+163

6　　200=19+181

7　　200=7+193

8　　200=3+197

D(200)=8

1992 年是一个偶数年,把 1992 年输入后则打印出 59 对(1+1)的素数对。兹录于下,以便对照。

1　　1992=983+1009

2　　1992=971+1021

3　　1992=953+1039

4　　1992=941+1051

5　　1992=929+1063

6　　1992=883+1109

7　　1992=863+1129

8　　1992=839+1153

9　　1992=829+1163

10　　1992=821+1171

11 1992=811+1181

12 1992=769+1223

13 1992=761+1231

14 1992=743+1249

15 1992=733+1259

16 1992=709+1283

17 1992=701+1291

18 1992=691+130

19 1992=673+1319

20 1992=631+1361

21 1992=619+1373

22 1992=593+1399

23 1992=569+1423

24 1992=563+1429

25 1992=541+1451

26 1992=521+1471

27 1992=509+1483

28 1992=503+1489

29 1992=499+1493

30 1992=461+1531

31 1992=449+1543

32 1992=443+1549

33 1992=439+1553

34 1992=433+1559

35 1992=421+1571

36 1992=409+1583

37 1992=383+1609

38 1992=379+1613

39 1992=373+1619

40 1992=293+1699

41 1992=283+1709

42　　1992=271+1721

43　　1992=269+1723

44　　1992=251+1741

45　　1992=239+1753

46　　1992=233+1759

47　　1992=191+1801

48　　1992=181+1811

49　　1992=131+1861

50　　1992=113+1879

51　　1992=103+1889

52　　1992=79+1913

53　　1992=61+1931

54　　1992=59+1933

55　　1992=43+1949

56　　1992=41+1951

57　　1992=19+1973

58　　1992=13+1979

59　　1992=5+1987

D(1992)=59

2. 哥德巴赫奇数猜想

奇数猜想的数学表达式为：对于任一个不小于 9 的奇数 N 有

N=P1+P2+P3

对于奇数 15 有

15=3+5+7

15=5+5+5

用数论符号表示为

T(15)=2

哥德巴赫奇数猜想的原始程序见 No. 10—12。

```
5 REM No. 10—12
10 REM 哥德巴赫奇数猜想程序
```

```
20 INPUT "请送入奇数=";N
30 FOR X=3 TO N/3 STEP 2
40 FOR I=2 TO SQR(X)
50 IF X/I=INT(X/I) THEN 100
60 NEXT I
70 A=0:B=B+1:GOSUB 300
100 NEXT X
120 END
300 R=N-X
310 FOR Y=X TO R/2 STEP2
320 FOR I=2 TO SQR(Y)
330 IF Y/I=INT(Y/I) THEN 430
340 NEXT I
350 Z=R-Y
360 FOR I=2 TO SQR(Z)
370 IF Z/I=INT (Z/I)THEN 430
380 NEXT I
390 A=A+1:M=M+1
410 PRINT M;" ";B;" ";" ";N;"=";X;"+";Y;"+";Z
430 NEXT Y
440 RETURN
```


原始程序给出了 N=33 的运行结果

```
1    1    1    33=3+7+23
2    1    2    33=3+11+19
3    1    3    33=3+13+17
4    2    1    33=5+5+23
5    2    2    33=5+11+17
6    3    1    33=7+7+19
7    3    2    33=7+13+13
8    4    1    33=11+11+11
```

运行结果的第一列表示第几组解，T(33)=8；第二列表示 P1

的顺序号，1 表示 P1 的第一个数 3，2 表示 P1 的第二个数 5……；第三列表明一个确定的 P1 使用几次，如 P1＝3 使用 3 次，即有 3 组解，P1＝5 使用两次，说明含 5 的有两组解……。

与 No. 10—10 相比，该程序除多用了几个 SQR 函数外，其它大同小异，可直接移植。目标程序见 No. 10—13。

```
5 REM No. 10—13
10 INPUT "Inputan ODD number";N
20 FOR X=3 TO N/3 STEP 2
30 I=2:W=X:GOSUB 90:II=P
40 IF X MOD I=0 THEN 70
50 I=I+1: IF I<=II THEN 40
60 A=0:B=B+1:GOSUB 120
70 NEXT
80 END
90 P=0:K=2
100 IF K*K>W THEN P=K:RETURN
110 K=K+1:GOTO 100
120 R=N-X:Y=X
130 I=2:W=Y:GOSUB 90:II=P
140 IF Y MOD I=0 THEN 210
150 I=I+1:IF I<II THEN 140
160 Z=R-Y:I=2:W=Z:GOSUB 90:II=P
170 IF Z MOD I=0 THEN 210
180 I=I+1:IF I<=II THEN 170
190 A=A+1:M=M+1
200 PRINT M;B;A;N"="X"+"Y"+"Z
210 Y=Y+2:IF Y<=R/2 THEN 130
220 RETURN
```

运行程序，输入 33，将打印与 No. 10—12 相同的结果。

输入 55，将打印 T(55)＝14

1	1	1	55＝3＋5＋47
2	1	2	55＝3＋11＋41
3	1	3	55＝3＋23＋29
4	2	1	55＝5＋7＋43
5	2	2	55＝5＋13＋37
6	2	3	55＝5＋19＋31
7	3	1	55＝7＋7＋41
8	3	2	55＝7＋11＋37
9	3	3	55＝7＋17＋31
10	3	4	55＝7＋19＋29
11	4	1	55＝11＋13＋31
12	5	1	55＝13＋13＋29
13	5	2	55＝13＋19＋23
14	6	1	55＝17＋19＋19

若输入 99，则打印出 T(99)＝30

1	1	1	99＝3＋7＋89
2	1	2	99＝3＋13＋83
3	1	3	99＝3＋17＋79
4	1	3	99＝3＋23＋73
5	1	5	99＝3＋29＋67
6	1	6	99＝3＋37＋59
7	1	7	99＝3＋43＋53
8	2	1	99＝5＋5＋89
9	2	2	99＝5＋11＋83
10	2	3	99＝5＋23＋71
11	2	4	99＝5＋41＋53
12	2	5	99＝5＋47＋47
13	3	1	99＝7＋13＋79
14	3	2	99＝7＋19＋73
15	3	3	99＝7＋31＋61
16	4	1	99＝11＋17＋71

17	4	2	99=11+29+59
18	4	3	99=11+41+47
19	5	1	99=13+13=73
20	5	2	99=13+19+67
21	5	3	99=13+43+43
22	6	1	99=17+23+59
23	6	2	99=17+29+53
24	6	3	99=17+41+41
25	7	1	99=19+19+61
26	7	2	99=19+37+43
27	8	1	99=23+23+53
28	8	2	99=23+29+47
29	9	1	99=29+29+41
30	10	1	99=31+31+37

1993 是个奇数，若输入 1993 则会打印出 3182 组结果，即 T(1993)=3182，仅 P1=3 就有 42 组解，为便于对照，特录前 43 个解如下：

T(1993)=3182

1	1	1	1993=3+3+1987
2	1	2	1993=3+11+1979
3	1	3	1993=3+17+1973
4	1	4	1993=3+41+1949
5	1	5	1993=3+59+1931
6	1	6	1993=3+83+1907
7	1	7	1993=3+89+1901
8	1	8	1993=3+101+1889
9	1	9	1993=3+113+1877
10	1	10	1993=3+167+1823
11	1	11	1993=3+179+1811
12	1	12	1993=3+257+1733

13	1	13	1993=3+269+1721
14	1	14	1993=3+281+1709
15	1	15	1993=3+293+1697
16	1	16	1993=3+353+1637
17	1	17	1993=3+383+1607
18	1	18	1993=3+389+1601
19	1	19	1993=3+419+1571
20	1	20	1993=3+431+1559
21	1	21	1993=3+467+1523
22	1	22	1993=3+479+1511
23	1	23	1993=3+491+1499
24	1	24	1993=3+503+1487
25	1	25	1993=3+509+1481
26	1	26	1993=3+557+1433
27	1	27	1993=3+563+1427
28	1	28	1993=3+617+1373
29	1	29	1993=3+683+1307
30	1	30	1993=3+701+1289
31	1	31	1993=3+761+1229
32	1	32	1993=3+773+1217
33	1	33	1993=3+797+1193
34	1	34	1993=3+809+1181
35	1	35	1993=3+827+1163
36	1	36	1993=3+839+1151
37	1	37	1993=3+881+1109
38	1	38	1993=3+887+1103
39	1	39	1993=3+929+1061
40	1	40	1993=3+941+1049
41	1	41	1993=3+971+1019
42	1	42	1993=3+977+1013
43	1	1	1993=5+37+1951

……

若输入 1995 则有 T(1995)＝2029，若输入 1997 则打印 3309 个结果，有 T(1997)＝3309。

总之程序 No. 10—11、No. 10. 10—13 千真万确地、一个不漏地打印出奇、偶数猜想的结果。

小结：

在程序的移植中，遇到 SQR 函数时，若精度要求不高，可使用递推法，近似求得，凡遇到进行整除判断的语句可使用 MOD 运算符直接移植。

总之，移植程序如同临摹名画。评价临摹技能的高低，不是看你对画框描画的多么精采逼真，而是看你对画的绘制是否维妙维肖、乱真传神。移植程序也是如此，移植中要敢于删削，要勇于创新。不求形似，但求神化。只要能够再现原始程序的主要功能，即为成功。另外要广开思路、千方百计，选择最佳方案，充分发挥 F BASIC 语言的特点、优势，以求编出最好的移植程序，使其达到甚至超过原始程序的水平。

附录 1 《成龙救金凤》模块结构程序清单

```
5 REM CHENG LONG SAVE JIN FENG
10 REM MAIN PROGRAM
15 GOSUB 400
20 CLEAR:CLS:CGSET RND(2),RND(3):CGEN2
25 LOCATE 8,8:PRINT "START!!!"
30 SCREEN 0,1:PLAY "M0V12M1V12Y1T3:M0V9T3:M1V2T3":
   FF=1:GOSUB 800
35 SPRITE ON :PLAY "V15T1:V15T1"
40 IF SCR$(1,2)<>"  "GOTO 50
45 GOSUB 500
50 SCREEN 0,0:VIEW:PRINT"      TIME=              PRESS
   START"
55 K=STRIG(0):IF K=0 GOTO 55
60 PLAY "O3C1EE:O3E1FG"
65 GOSUB 550
70 GOSUB 620
75 TI=TI-1:LOCATE 7,0 :PRINT TI:"      "
80 IF TI=0 THEN 115
85 GOSUB 200
90 GOSUB 270
95 GOSUB 300
100 IF CRASH(7)>=0 THEN PLAY "O1B2AGFEDC":GOTO 115
105 GOTO 75
110 GOSUB 700:GOTO 130
115 CLS:LOCATE 9,6:PRINT "GAME  OVER!"
120 SPRITE OFF
130 LOCATE 2,20:PRINT "CONTINUE/start END/select"
135 S=STRIG(0): IF S=0 THEN 135
140 IF S=1 THEN RUN 20
145 IF S=2 THEN CLS:LOCATE 10,4:PRINT "GOOD BYE!!!":
```

```
    LOCATE 0,21:END
150 GOTO 135
200 REM CONTROL
205 S=0:SX=0:SY=0
210 K=STICK(0):IF K=0 GOTO 265
215 IF K=1 THEN S=3:SX=1
220 IF K=2 THEN S=7:SX=-1
225 IF K=4 THEN S=5:SY=1
230 IF K=8 THEN S=1:SY=-1
235 PX=(XPOS(7)-12)/8:PY=(YPOS(7)-16)/8
240 S$=SCR$(PX+SX,PY+SY+1):S1$=SCR$(PX+SX,PY
    +SY)
245 IF S=0 OR S$=" "GOTO 265
250 DEF MOVE(7)=SPRITE(0,S,1,4,0,RND(4)):POSITION 7,
    PX*8+12,PY*8+16:MOVE 7
255 IF S=3 OR S=7 THEN PLAY "O0#F0RO1#F":GOTO 265
260 PLAY "O1#F0RO0#FR"
265 RETURN
270 OPEN BOX
275 IF S1$<>MID$(PS$,PS,1) THEN 295
280 PLAY "O4C0#CD#DEF#FG#G"
285 LOCATE PX+SX,PY+SY:PRINT " ":T=T+10:LOCATE 8,
    22:PRINT T
290 PS=PS+1: IF PS>PL THEN PLAY "O4C3DEFG":
    O4E3FGAB": GOTO 110
295 RETURN
300 REM BOGY MOVE
305 F=F+1: IF F>KN-1 THEN F=0
310 KD=VCT(F):IF KD>0 GOTO 320
315 KD=E(F)
320 KX=(XPOS(F)-12)/8:JX=SGN(PX-KX)
```

```
325 KY=(YPOS(F)-16)/8:JY=SGN(PY-KY-1)
330 SK$=SCR$(KX,KY):SR$=SCR$(KX+1,KY):SL$=SCR
    $(KX-1,KY):SD$=SCR$(KX,KY+1):SU$=SCR$(KX,
    KY-1)
335 IF (SR$+SL$<>" ") AND (SD$+SU$<>" ") GOTO 345
340 IF (KD=3 AND JX=1) OR (KD=7 AND JX=-1) OR (KD=5
    AND JY=1) OR (KD=1 AND JY=-1) GOTO 390
343 KD=KD+4+8*(KD>3): GOTO 390
345 IF JX*JY<>0 AND RND(2)=1 GOTO 355
350 IF JX=1 AND SR$<>" " THEN KD=3: GOTO 390
353 IF JX=-1 AND SL$<>" " THEN KD=7: GOTO 390
355 IF JY=1 AND SD$<>" " THEN KD=5: GOTO 390
360 IF JY=-1 AND SU$<>" " THEN KD=1: TOGO 390
365 S=RND(4)+1: ON S GOTO 370,375,380,385
370 IF SR$<>" " THEN KD=3: GOTO 390
375 IF SL$<>" " THEN KD=7: GOTO 390
380 IF SD$<>" " THEN KD=5: GOTO 390
385 IF SU$<>" " THEN KD=1
390 DEF MOVE(F)=SPRITE(11,KD,1,4,0,RND(4)):POSITION
    F,KX*8+12,KY*8+16:MOVE F
393 E(F)=KD
395 RETURN
400 REM BEGIN PICTURE
405 CGSET 0,1: SPRITE ON :CLS:PALETB 0,8,48,47,47
410 FOR I=0 TO 27: LOCATE I,12:PRINT CHR$(197): NEXT
415 LOCATE 10,2:PRINT CHR$(216);CHR$(218):LOCATE 8,
    3:PRINT CHR$(216);CHR$(217);CHR$(218)
420 LOCATE 1,8:PRINT CHR$(208):LOCATE 0,9:PRINT CHR
    $(249);CHR$(203);CHR$(251):LOCATE 1,10:PRINT CHR
    $(238):LOCATE 1,11:PRINT CHR$(212)
425 LOCATE 2,16:PRINT "CHENG LONG SAVES JIN FENG":LO-
```

```
    CATE 8,20:PRINT "BY        YU CHUN"
430 DEF MOVE(4)=SPRITE (0,7,8,90,0,3):POSITION 4,210,
    100:MOVE 4
435 DEF MOVE (5)=SPRITE(1,7,8,80,0,3):POSITION 5,220,
    104:MOVE 5
440 PLAY"M0V12MIV12YIT2:M0V9T2:MIV2T2":FF=1:GOSUB
    800
445 DEF MOVE(6)=SPRITE(15,5,1,23,1,0):POSITION 6,88,48:
    MOVE  6
450 IF CRASH(6)<>5 THEN 450
455 DEF MOVE(5)=SPRITE(1,2,1,58,0,3):POSITION 5,70,104:
    MOVE  5
460 DEF MOVE(6)=SPRITE(15,2,1,55,0,0):POSITION 6,76,96:
    MOVE  6
465 PALETB 0,10,48,47,47
470 DEF MOVE(4)=SPRITE(0,3,1,94,0,3):POSITION 4,60,100:
    MOVE  4
475 GOSUB 900:ERA 4
480 RETURN
500 REM BUILDING
505 B$=CHR$(187):J$=CHR$(210):J$=J$+"      "+J$
510 FOR X=1 TO 21 STEP 4
515 FOR Y=2 TO 20 STEP 3
520 LOCATE X,Y:PRINT B$;B$;B$;B$: IF Y>19 THEN 530
525 LOCATE X,Y+1:PRINT J$:LOCATE X,Y+2:PRINT J$
530 NEXT:NEXT
535 FOR I=2 TO 20 STEP 3
540 LOCATE 25,I:PRINT B$:NEXT
545 RETURN
550 REM BOX
555 P1=RND(3)+5
```

```
560 C=80:IF P1=7 THEN C=90
565 P2=P1*22+C:PS$=CHR$(P2)
570 FOR I=1 TO 8
575 PS$=PS$+CHR$(P2+I):NEXT
580 PL=LEN(PS$):PS=1
585 FOR S=1 TO PL
590 SX=2+RND(24):SY=2+RND(18)
595 S$=SCR$(SX,SY+1)
600 IF (S$=" ") OR (INSTR(PS$,S$)>0) OR ((SX-1) MOD 4
    =0) THEN 590
605 LOCATE SX,SY:PRINT MID$(PS$,S,1)
610 NEXT
615 RETURN
620 REM SHOW SPRITE
625 LOCATE 12,0:PRINT"                              "
630 DEF SPRITE 3,(2,1,0,0,0)="$%&'":SPRITE 3,200,16
635 T=0:F=0:KN=2:KX=0:KY=0:KD=0:TI=800
640 LOCATE 0,22:PRINT "     SCORE  ";T
645 DEF MOVE(7)=SPRITE(0,0,2,4,0,RND(4)):POSITION 7,
    12,168:MOVE 7
650 DIM E(KN-1)
655 FOR F=0 TO KN-1
660 DEF MOVE(F)=SPRITE(11,1,2,4,0,RND(4))
665 SX=10+RND(17):SY=1+RND(10)
670 IF SCR$(SX,SY)=" " OR SCR$(SX,SY-1)=" " GOTO 665
675 POSITION F,SX*8+12,SY*8+16:MOVE F
680 NEXT
685 RETURN
700 REM VICTORY 1
705 ERA 0,1,2,3,4,5,6,7:SPRITE 3:CLS
710 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(208);:NEXT
```

```
715 PALETB 0,10,10,21,49
720 FOR I=2 TO 3
725 DEF MOVE(I)=SPRITE(I-2,4*I-5,2,58,0):POSITION I,(I
    -2)*240,104:MOVE I:NEXT
730 PLAY "M0V12M1V9Y2T1:M0V12T1:M1V0T1":FF=0:GUSUB
    800
750 REM VICTORY 2
755 ERA 2,3:CLS:PALETB 0,5,22,39,56
760 FOR I=0 TO 27:LOCATE I,12:PRINT CHR$(197);:NEXT
765 FOR I=2 TO 3
770 DEF MOVE(I)=SPRITE(I-2,7,4,230,0)
775 POSITION I,(I-3)*8+230,104:MOVE I:NEXT
780 PLAY:M0V12M1V12Y1T4:M0V9T4:M1V2T4":FF=1:GOSUB
    800
785 RETURN
800 REM MUSIC 1
805 PLAY "O4G5E3GA4G1:O4C5O3GA:O1C3CC5C3C"
810 PLAY "E3DCDE4D1:GDG:C5C3CC5"
815 PLAY "E3G7G3E4G1:AGGO4C:C3CC5C3CC5"
820 PLAY "A3O5C6O4G3A:DCO3G:C3CC5C3C"
825 PLAY "DE1DC7:CEC:C5C3CC5"
830 PLAY "G5E3GA4G1:O4CO3GA:G3GG5G3G"
835 PLAY "E3DCDE4D1:GDG:G5G3GG5"
840 PLAY "E3G7G3E4G1:AGGO4C:G3GG5C3CC5"
845 PLAY "A3O5C6D3G:ECD:C3CC5C3C"
850 PLAY "DE1DC7:GDC:C5C3CC5"
855 IF FF=0 THEN FF=1:GOTO 805
860 RETURN
900 REM MUSIC 2
905 PLAY "M0V10M1V8Y3T2:M0V3T2:T2"
910 PLAY "O4C3CO3GA:O3C3CO2GA:O2C5G"
```

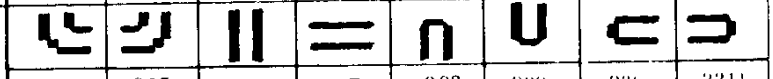
```
915 PLAY "O4CCO3GADG6:O3CCO2GADG6:CGDG"
920 PLAY "D3G6A3O4E6:D3G6A3O3E6:DGAO3E"
925 PLAY "O3A3O4E6:O2A3O3E6:O2AO3E"
930 PLAY "C3CO3GA:C3CO2GA:O2CG"
935 PLAY "O4CCO3GA:O3CCO2GA:CG"
940 RETURN
```

附录 2 （见封二）

附录 3 （见封四）

附录 4 （见封三）

附录 5　背景图形与 ASCII 码对照表

	1	2	3	4	5	6	7	8
A								
	0	1	2	3	4	5	6	7
B								
	8	9	10	11	12	13	14	15
C								
	16	17	18	19	20	21	22	23
D								
	24	25	26	27	28	29	30	31
E								
	184	185	186	187	188	189	190	191
F								
	192	193	194	195	196	197	198	199
G								
	200	201	202	203	204	205	206	207
H								
	208	209	210	211	212	213	214	215
I								
	216	217	218	219	220	221	222	223
J								
	224	225	226	227	228	229	230	231
K								
	232	233	331	235	236	237	238	239
L								
	240	241	242	243	244	245	246	247
M								
	248	249	250	251	252	253	254	255

附录 6　卡通编号表

编号	卡通名称	编号	卡通名称
0	玛丽	8	太空杀手卫星
1	丽莎	9	太空船
2	苍蝇	10	爆炸
3	飞鱼	11	妖怪
4	企鹅	12	雷射
5	火球	13	乌龟
6	车	14	螃蟹
7	太空保垒	15	鸟

附录 7　符号代码表 A(主要用于 sp 面)

请参阅本书封底卡通图形库中每个卡通图案 4 角的 10 进制数字

10 进制代码	16 进制代码	说　明	10 进制代码	16 进制代码	说　明	10 进制代码	16 进制代码	说　明
0	00	玛丽(走 1)	21	15	玛丽(背面)	42	2A	(跳)
1	01		22	16		43	2B	
2	02		23	17		44	2C	丽莎(滑)
3	03		24	18	玛丽(跌)	45	2D	
4	04	玛丽(走 2)	25	19		46	2E	
5	05		26	1A		47	2F	
6	06		27	1B		48	30	丽莎(背面)
7	07		28	1C	丽莎(走 1)	49	31	
8	08	玛丽(走 3)	29	1D		50	32	
9	09		30	1E		51	33	
10	0A		31	1F		52	34	丽莎(跌)
11	0B		32	20	丽莎(走 2)	53	35	
12	0C	玛丽(跳)	33	21		54	36	
13	0D		34	22		55	37	
14	0E		35	23		56	38	苍蝇(1)
15	0F		36	24	丽莎(走 3)	57	39	
16	10	玛丽(滑)	37	25		58	3A	
17	11		38	26		59	3B	
18	12		39	27		60	3C	苍蝇(2)
19	13		40	28	丽莎	61	3D	
20	14		41	29		62	3E	

附录 7 （续）

10 进制代码	16 进制代码	说明
63	3F	
64	40	飞鱼（1）
65	41	
66	42	
67	43	
68	44	飞鱼（2）
69	45	
70	46	
71	47	
72	48	飞鱼（左上 1）
73	49	
74	4A	
75	4B	
76	4C	飞鱼（左上 2）
77	4D	
78	4E	
79	4F	
80	50	飞鱼（前 1）
81	51	
82	52	
83	53	
84	54	飞鱼（前 2）
85	55	
86	56	
87	57	
88	58	妖怪（1）
89	59	
90	5A	
91	5B	
92	5C	妖怪（2）
93	5D	
94	5E	
95	5F	
96	60	企鹅（走 1）
97	61	
98	62	
99	63	
100	64	企鹅（走 2）
101	65	
102	66	
103	67	
104	68	企鹅（正面）
105	69	
106	6A	
107	6B	
108	6C	企鹅（背面）
109	6D	
110	6E	
111	6F	
112	70	
113	71	火球（1）
114	72	
115	73	
116	74	火球（2）
117	75	
118	76	
119	77	
120	78	车（左 1）
121	79	
122	7A	
123	7B	
124	7C	车（左 2）
125	7D	
126	7E	
127	7F	
128	80	车（左上 1）
129	81	
130	82	
131	83	
132	84	车（左上 2）
133	85	
134	86	
135	87	
136	88	车
137	89	

附录 7 （续）

10 进制代码	16 进制代码	说　明
138	8A	（上 1）
139	8B	
140	8C	车（上 2）
141	8D	
142	8E	
143	8F	
144	90	太空站（1）
145	91	
146	92	
147	93	
148	94	太空站（2）
149	95	
150	96	
151	97	
152	98	杀手卫星（左）
153	99	
154	9A	
155	9B	
156	9C	杀手卫星（左上）
157	9D	
158	9E	
159	9F	
160	A0	杀手卫星（上）
161	A1	
162	A2	
163	A3	
164	A4	太空船（左）
165	A5	
166	A6	
167	A7	
168	A8	太空船（左上）
169	A9	
170	AA	
171	AB	
172	AC	太空船（上）
173	AD	
174	AE	
175	AF	
176	B0	爆炸（开始）
177	B1	
178	B2	
179	B3	
180	B4	爆炸（散发）
181	B5	
182	B6	
183	B7	
184	B8	乌龟走（1）
185	B9	
186	BA	
187	BB	
188	BC	乌龟立正（2）
189	BD	
190	BE	
191	BF	
192	C0	螃蟹（1）
193	C1	
194	C2	
195	C3	
196	C4	螃蟹（2）
197	C5	
198	C6	
199	C7	
200	C8	飞鸟（1）
201	C9	
202	CA	
203	CB	
204	CC	飞鸟（2）
205	CD	
206	CE	
207	CF	
208	D0	激光弹（1），（2）
209	D1	
210	D2	激光束（1），（2）
211	D3	
212	D4	激光刀

附录 7 （续）

10 进制代码	16 进制代码	说明	10 进制代码	16 进制代码	说明	10 进制代码	16 进制代码	说明
213	D5	(1),(2)	228	E4	笔	243	F3	4
214	D6	SP 用	229	E5		244	F4	C
215	D7	配色号码	230	E6		245	F5	O
216	D8	(1)(2)(3)	231	E7		246	F6	M
217	D9	音乐板用	232	E8	向下羽毛笔	247	F7	P
218	DA	三音程	233	E9		248	F8	U
219	DB	(1)(2)(3)	234	EA		249	F9	T
220	DC	光标(1)	235	EB		250	FA	E
221	DD	光标(2)	236	EC		251	FB	R
222	DE	指示器(1)	237	ED		252	FC	O
223	DF	指示器(2)	238	EE		253	FD	P
224	E0	向	239	EF		254	FE	R
225	E1	上	240	F0	1	255	FF	A
226	E2	羽	241	F1	2			T
227	E3	毛	242	F2	3			E
								R

附录 8　符号代码表 B(用于 BG 面)

键盘的字符,BG GRAPHIC 所使用的符号。0～31,184～255 是 BG 面所使用的各种图形,其余代表字母、符号、假名等。

10 进制代码	16 进制代码	对应的符号	10 进制代码	16 进制代码	对应的符号	10 进制代码	16 进制代码	对应的符号
0	00	A1	21	15	C6	42	2A	*
1	01	A2	22	16	C7	43	2B	+
2	02	A3	23	17	C8	44	2C	,
3	03	A4	24	18	D1	45	2D	−
4	04	A5	25	19	D2	46	2E	.
5	05	A6	26	1A	D3	47	2F	/
6	06	A7	27	1B	D4	48	30	0
7	0:	A8	28	1C	D5	49	31	1
8	08	B1	29	1D	D6	50	32	2
9	09	B2	30	1E	D7	51	33	3
10	0A	B3	31	1F	D8	52	34	4
11	0B	B4	32	20	空格	53	35	5
12	0C	B5	33	21	!	54	36	6
13	0D	B6	34	22	"	55	37	7
14	0E	B7	35	23	#	56	38	8
15	0F	B8	36	24	$	57	39	9
16	10	C1	37	25	%	58	3A	:
17	11	C2	38	26	&	59	3B	;
18	12	C3	39	27	'	60	3C	<
19	13	C4	40	28	(	61	3D	=
20	14	C5	41	29	)	62	3E	>

附录 8 （续）

10 进制代码	16 进制代码	对应的符号	10 进制代码	16 进制代码	对应的符号	10 进制代码	16 进制代码	对应的符号
63	3F	?	88	58	X	113	71	q
64	40	@	89	59	Y	114	72	r
65	41	A	90	5A	Z	115	73	S
66	42	B	91	5B	[	116	74	t
67	43	C	92	5C	¥	117	75	u
68	44	D	93	5D	]	118	76	v
69	45	E	94	5E	^	119	77	w
70	46	F	95	5F	—	120	78	x
71	47	G	96	60	'	121	79	y
72	48	H	97	61	a	122	7A	z
73	49	I	98	62	b	123	7B	{
74	4A	J	99	63	c	124	7C	\
75	4B	K	100	64	d	125	7D	}
76	4C	L	101	65	e	126	7E	⌴
77	4D	M	102	66	f	127	7F	⌐
78	4E	N	103	67	g	128	80	一
79	4F	O	104	68	h	129	81	二
80	50	P	105	69	i	130	82	三
81	51	Q	106	6A	j	131	83	四
82	52	R	107	6B	k	132	84	五
83	53	S	108	6C	l	133	85	六
84	54	T	109	6D	m	134	86	七
85	55	u	110	6E	n	135	87	八
86	56	v	111	6F	o	136	88	九
87	57	W	112	70	p	137	89	十

附录 8 （续）

10 进制代码	16 进制代码	对应的符号	10 进制代码	16 进制代码	对应的符号	10 进制代码	16 进制代码	对应的符号
						188	BC	E5
138	8A		163	A3	ッ	189	BD	E6
139	8B		164	A4	テ	190	BE	E7
140	8C		165	A5	ド	191	BF	E8
141	8D		166	A6	バ	192	C0	F1
142	8E		167	A7	ビ	193	C1	F2
143	8F		168	A8	ブ	194	C2	F3
144	90		169	A9	ベ	195	C3	F4
145	91		170	AA	ボ	196	C4	F5
146	92		171	AB	パ	197	C5	F6
147	93		172	AC	ピ	198	C6	F7
148	94		173	AD	プ	199	C7	F8
149	95		174	AE	ペ	200	C8	G1
150	96		175	AF	ポ	201	C9	G2
151	97	ガ	176	B0	□	202	CA	G3
152	98	ギ	177	B1	°	203	CB	G4
153	99	グ	178	B2	‘	204	CC	G5
154	9A	ゲ	179	B3	’	205	CD	G6
155	9B	ゴ	180	B4	C	206	CE	G7
156	9C	ザ	181	B5	×	207	CF	G8
157	9D	ジ	182	B6	÷	208	D0	H1
158	9E	ズ	183	B7	）	209	D1	H2
159	9F	ゼ				210	D2	H3
160	A0	ゾ	184	B8	E1	211	D3	H4
161	A1	ダ	185	B9	E2	212	D4	H5
162	A2	ヂ	186	BA	E3			
			187	BB	E4			

玛丽兄弟用

风景

星

球

金刚二代用

附录 8 （续）

10 进制代码	16 进制代码	对应的符号		10 进制代码	16 进制代码	对应的符号		10 进制代码	16 进制代码	对应的符号	
213	D5	H6	金刚二代	228	E4	J5	双图表线	243	F3	L4	平板图
214	D6	H7		229	E5	J6		244	F4	L5	
215	D7	H8		230	E6	J7		245	F5	L6	
216	D8	I1	岛	231	E7	J8		246	F6	L7	
217	D9	I2		232	E8	K1	单图表线	247	F7	L8	
218	DA	I3		233	E9	K2		248	F8	M1	
219	DB	I4		234	EA	K3		249	F9	M2	结构用
220	DC	I5		235	EB	K4		250	FA	M3	
221	DD	I6	双图表线	236	EC	K5		251	FB	M4	
222	DE	I7		237	ED	K6		252	FC	M5	
223	DF	I8		238	EE	K7		253	FD	M6	方块用
224	E0	J1		239	EF	K8		254	FE	M7	
225	E1	J2		240	F0	L1		255	FF	M8	
226	E2	J3		241	F1	L2					
227	E3	J4		242	F2	L3					

附录 9　五十二种颜色代码表

	16进制	10进制	16进制	10进制	16进制	10进制	16进制	10进制
灰白系	00	0	10	16	20	32	30	48
兰系	01	1	11	17	21	33	31	49
	02	2	12	18	22	34	32	50
	03	3	13	19	23	35	33	51
	04	4	14	20	24	36	34	52
红系	05	5	15	21	25	37	35	53
	06	6	16	22	26	38	36	54
	07	7	17	23	27	39	37	55
	08	8	18	24	28	40	38	56
绿系	09	9	19	25	29	41	39	57
	0A	10	1A	26	2A	42	3A	58
	0B	11	1B	27	2B	43	3B	59
	0C	12	1C	28	2C	44	3C	60
黑系	0D	13	1D	29	2D	45		
	0E	14	1E	30	2E	46		
	0F	15	1F	31	2F	47		

灰~白

过渡色

黑色

暗←——————————→明

附录 10　V3.10 出错代码表

错误代码	错误表示	错误信息	说明
0	NF	NEXT without FOR	无 FOR 却有 NEXT
1	SN	Syntax error	语法错误
2	RG	RETURN without GOSUB	无 GOSUB 却有 RETURN
3	OD	out of DATA	READ 读的资料未备于 DATA 中
4	IL	Illegal function call	函数的称呼法错误
5	OV	Overflow	演算结果超过许可的范围
6	OM	Out of memory	内存不足
7	UL	Undefined line Number	无 GOTO、GOSUB、IF 所指的行号
8	SO	Subscript out of range	数组下标超出规定
9	DD	Duplicate Definition	数组双重定义
10	DZ	Division by zero	用 0 作除数
11	TM	Type mismatch	变量的类型不一致
12	ST	String too long	字串超过 31 个字符
13	FT	Formula too complex	式子过于复杂，如()太多
14	CC	Can't continue	CONT 无法继续执行程序
15	UF	Undefind Function	使用了未定义的函数
16	MO	Missing oprand	指令未指定所需要的参数
17	TP	Tape read error	无法从磁带读取资料
18	NR	No RESUME	错误处理子程序无 RESUME
19	RE	RESUME without error	无出错转移语句却遇到 RESUME
20	NB	No BGGET	无 BGGET 却遇到 BGPUT
21	UP	Unprintable error	遇到无法显示打印的错误代码

附录 11　BS. 2A 版本保留字及代码表

保留字	10 进制代码	16 进制代码	保留字	10 进制代码	16 进制代码
ABS	202	CA	LEFT $	224	E0
AND	241	F1	LINPUT	144	90
ASC	203	CB	LIST	134	86
BEEP	164	A4	LOAD	165	A5
CALL	173	AD	LOCATE	174	AE
CGEN	171	AB	MID $	212	D4
CGSET	159	9F	MOD	251	FB
CHR $	220	DC	MOVE	161	A1
CLEAR	153	99	NEXT	141	8D
CLS	152	98	NEW	157	9D
COLOR	169	A9	NOT	242	F2
CONT	151	97	OFF	155	9B
CSRLIN	219	DB	ON	154	9A
CUT	156	9C	OR	240	F0
DATA	145	91	PALET	175	AF
DEF	170	AA	PAUSE	142	8E
DIM	148	94	PEEK	207	CF
END	162	A2	PLAY	163	A3
ERA	176	B0	POKE	158	9E
FOR	140	8C	POS	218	DA
FRE	205	CD	POSITION	167	A7
GOTO	128	90	PRINT	139	8B
GOSUB	129	81	READ	147	93
HEX $	221	DD	REM	149	95
IF	146	92	RESTORE	132	84
INPUT	143	8F	RETURN	131	83
INKEY $	222	DE	RIGHT	223	DF
KEY	168	A8	RND	208	D0
LEN	206	CE	RUN	130	82

附录 11 （续表）

保留字	10 进制代码	16 进制代码	保留字	10 进制代码	16 进制代码
SAVE	166	A6	VAL	217	D9
SGN	209	D1	VIEW	160	A0
SPC	210	D2	XPOS	215	D7
SPRITE	138	8A	XOR	239	EF
SCR $	225	E1	YPOS	216	D8
STICK	213	D5	<>	243	F3
STEP	137	89	>=	244	F4
STOP	150	96	<=	245	F5
STRIG	214	D6	=	246	F6
STR $	204	CC	>	247	F7
SWAP	172	AC	<	248	F8
SYSTEM	135	87	+	249	F9
TAB	211	D3	−	250	FA
TO	136	88	/	252	FC
THEN	133	85	*	253	FD

附录 12　V3.10 版本保留字及代码表

保留字	10 进制代码	16 进制代码	保留字	10 进制代码	16 进制代码
ABS	202	CA	FOR	140	8C
AND	241	F1	FILTER	184	B8
ASC	203	CB	FIND	178	B2
AUTO	181	B5	REE	205	CD
BACKUP	187	B9	GAME	179	B3
BEEP	164	A4	GOSUB	129	81
BGGET	191	BF	GOTO	128	80
BGPUT	190	BE	HEX $	221	DD
BGTOOL	180	B4	IF	146	92
CALL	173	AD	INKEY $	222	DE
CAN	192	C0	INPUT	143	8F
CGEN	171	AB	INSTR	226	E2
CGSET	159	9F	KEY	168	A8
CHR $	220	DC	LEFT $	224	E0
CLEAR	153	99	LEN	206	CE
CLS	152	98	LINPUT	144	90
CLICK	185	B9	LIST	134	86
COLOR	169	A9	LOAD	165	A5
CONT	151	97	LOCATE	174	AE
CRASH	227	E3	MID $	212	D4
CSRLIN	219	DB	MOD	251	FB
CUT	156	9C	MOVE	161	A1
DATA	145	91	NEXT	141	8D
DEF	170	AA	NEW	157	9D
DELETE	182	B6	NOT	242	F2
DIM	148	94	OFF	155	9B
END	162	A2	ON	154	9A
ERA	176	B0	OR	240	F0
ERL	229	E5	PALET	175	AF
ERR	228	E4	PAUSE	142	8E
ERROR	188	BC	PEEK	207	CF

附录 12 （续）

保留字	10 进制代码	16 进制代码	保留字	10 进制代码	16 进制代码
PALY	163	A3	STRIG $	214	D6
POKE	158	9E	SWAP	172	AC
POS	218	DA	SYSTEM	135	87
POSITION	167	A7	TAB	211	D3
PRINT	139	8B	THEN	133	85
READ	147	93	TO	136	88
REM	149	95	TR	177	B1
RENUM	183	B7	VAL	217	D9
RESTORE	132	84	VCT	230	E6
RESUME	189	BD	VIEW	160	A0
RETURN	131	83	XPOS	215	D7
RIGHT $	223	DF	XOR	239	EF
RND	208	D0	YPOS	216	D8
RUN	130	82	〈〉	243	F3
SVE	166	A6	〉=	244	F4
SCR $	225	E1	〈=	245	F5
SCREEN	186	BA	=	246	F6
SGN	209	D1	〈	247	F7
SPC	210	D2	〉	248	F8
SPRITE	138	BA	+	249	F9
STEP	137	89	-	250	FA
STICK	213	D5	/	252	FC
STOP	150	96	*	253	FD
STR $	204	CC			

附录 13　BS・2A 指令汇总表

13－1　命令一览表

命　　令	功　　能
CLEAR &H××××	规定 BASIC 的使用范围
NEW	清除内存
LIST [[m],[－n]]	将内存的程序按行号大小列表
RUN [n]	执行内存中的程序
CONT	使中断的程序继续往下执行
LOAD ["文件名"]	从磁带中读出程序
SAVE["文件名"]	把程序写入磁带中
LOAD? ["文件名"]	核对待保存程序是否正确地写入磁带中
CALL ××××	调用机器语言子程序
PRINT [],[]	打印计算结果或数据资料
[]；[]	把 16 进制数转换成 10 进制数
KEY n,"字符串"	设定功能键
KEY LIST	显示各功能键的定义

13－2　一般语句一览表

语　　句	功　　能
PRINT []；[] [],[]	打印计算结果或数据资料
INPUT ["提示信息"] {；}变量，…	人机对话，从键盘上输入数据资料
LINPUT["提示信息"] {；}字符变量	读取一整行数据，包括段落分隔符号在内
CLEAR	清除内存中所有变量的值
DIM 变量名(m1,[m2])	定义数组
GOTO n	无条件转移到由 n 指定的行号
GOSUB n	调用指定行号的子程序
RETURN	由子程序返回到主程序

13—2 （续）

语　　句	功　　能
IF 表达式 THEN {行号 / 语句}	条件转移，可用来判断并执行某种功能
FOR ～ TO ～ STEP NEXT	循环语句
ON 表达式 {GOTO / GOSUB / RETURN / RESTORE} 行号 1，行号 2，……	控制转向指令
STOP	暂停指令
END	程序结束指令
SWAP 变量 1，变量 2	交换两个变量的内容
REM [叙述]	注释指令
READ 变量 [，变量，……]	读取数据（从 DATA 中读取）
DATA 数据 [，数据，……]	置数据指令（供 READ 执行）
RESTORE[行号]	恢复数据区指令
POKE 地址，数据	将数据存入指定地址的内存单元中去
BEEP	使扬声器发出蜂鸣声
PLAY“弦乐资料”	演奏音乐
PAUSE n	运行中暂停指令

13—3　一般函数指令一览表

函　　数	功　　能
ABS(X)	对 X 取绝对值
SGN(X)	判别 X 的符号
RND(X)	产生 X 以内的随机整数
ASC(字符)	将字符转换成 ASCII 码
CHR $ (X)	求 ASCII 码 X 对应的字
VAL(字符)	将字符串转换成 10 进制数
STR $ (X)	将 10 进制数转换民字符串
HEX $ (X)	将 10 进制数转换成 16 进制数

13—3 （续）

函　数	功　能
LEFT $(字符串,n)	从字符串左边取 n 个字符
RIGHT $(字符串,n)	从字符串右边取 n 个字符
MID $(字符串,N,n)	从字符串的第 N 位取 n 个字符
LEN(字符串)	计算字符串的长度
PEEK(X)	取指定内存地址 X 中的数值

13—4 特殊函数指令一览表

函　数	功　能
POS(0)	了解光标在目前画面在目前上的水平位置
CSRLIN	给出光标在垂直方向的位置座标
FRE	测试内存单元的空余数
STICK(X)	读操纵器的方向键值
STRIG(X)	读操纵器的功能键值
SCR $(X,Y,SW)	求出 BG 面(X,Y)处图案的数值
INKEY $[0]	读取键盘输入的一个值
TAB	(待定)
SPC	(待定)

13—5 画面控制指令一览表

指　令	功　能
LOCATE X,Y	设定显示位置的座标(X,Y)
COLOR X,Y,n	在屏幕上(X,Y)位置染 n 号颜色
CGEN (n)	决定 BG 面、SP 面上符号的分配
CGSET[m][,n]	规定 BG、SP 面上的颜色组合
CLS	清屏
PALETB n,c1,c2,c3,c4	定义 BG 面颜色代码
PALETS n,c1,c2,c3,c4	定义 SP 面颜色代码
SYSTEM	从 BASIC 状态返回选择菜单
VIEW	调出 BG GRAPHIC 画面
DEF MOVE (n) =SPRITE	定义动作 SP

13—5 （续）

函　数	功　能
(A,B,C,D,E,F)	
MOVE n0[,n1,……;]	启动 SP 运动
CUT n0[,n1,……]	使运动中的 SP 停止
ERA n0[,n1,……]	令运动中的 SP 停止并消失
POSITION n,X,Y	规定 n 号动作 SP 的起点座标(X,Y)
XPOS(n)	求出 n 号动作 SP 的水平座标
YPOS(n)	求出 n 号动作 SP 的垂直座标
MOVE(n)	判断 n 号动作 SP 的动作是否结束
DEF SPRITE n,(A,B,C,D,E)=字符表达式	定义 n 号显示 SP
SPRITE n,X,Y	定义 n 号显示 SP 的显示座标(X,Y)
SPRITE n	令 n 号显示 SP 消失
SPRITE ON	开 SP 显示
SPRITE OFF	关 SP 显示

附录 14　V3.10 指令汇总表

14－1　一般命令一览表

命　令	功　能
CLEAR &H××××	规定 BASIC 的使用范围
NEW	清除内存
LIST [[m],[－n]]	将内存的程序按行号大小列表
RUN [n]	执行内存中的程序
CONT	使中断的程序继续往下执行
LOAD ["文件名"]	从磁带中读出程序
SAVE["文件名"]	把程序写入磁带中
LOAD? ["文件名"]	核对待保存程序是否正确地写入磁带中
CALL ××××	调用机器语言子程序
PRINT [],[] [];[]	1.打印计算结果或数据资料 2.把 16 进制数转换成 10 进制数
KEY n,"字符串"	设定功能键
KEY LIST	显示各功能键的定义
FIND"字符串"	从程序中寻找指定的字符串
GAME n	从内存中调入 n 号游戏程序
AUTO [n][,m]	自动提供行号
DELETE [行号 1][－行号 2]	删除部分程序
RENUM [新行号][,现行号][,增量]	重编程序语句行号
BGTOOL	调出 BG GRAPHIC 画面

14－2　一般语句一览表

语　句	功　能
PRINT [];[] [],[]	打印计算结果或数据资料
INPUT ["提示信息"] {;}变量,……	人机对话,从键盘上输入数据资料
LINPUT["提示信息"] {;}字符变量	读取一整行数据,包括段落分隔符号在内

14—2 （续）

语　句	功　能
CLEAR	清除内存中所有变量的值
DIM 变量名(m1[,m2])	定义数组
……	
GOTO n	无条件转移到由 n 指定的行号
GOSUB n	调用指定行号的子程序
RETURN	由子程序返回到主程序
IF 表达式 THEN {行号 / 语句}	条件转移，可用来判断并执行某种功能
FOR ～ TO ～ STEP	循环语句
NEXT	
ON 表达式 {GOTO / GOSUB / RETURN / RESTORE}	控制转向指令
行号 1,行号 2,……	
STOP	暂停指令
END	程序结束指令
SWAP 变量 1,变量 2	交换两个变量的内容
REM [叙述]	注释指令
READ 变量[,变量……]	从 DATA 中读取数据
DATA 数据[,数据……]	置数据指令，供 READ 执行
RESTORE[行号]	恢复数据区指令
POKE 地址,数据	将数据存入指定地址的内存单元中去
BEEP	使扬声器发出蜂鸣声
PLAY "弦乐资料"	演奏音乐
PAUSE n	运行中暂停指令
SCREEN n1,n2	设置屏幕的显示模式
TRON	跟踪程序运行
TROFF	解除跟踪
BGPUT	(待定)
BGGET	(待定)

14—2 （续）

语　句	功　能
CLICK ON	（待定）
CLICK OFF	（待定）
FILTER n	滤色指令
BACKUP	磁盘备份指令
SYSTEM	打开系统指令

14—3 一般函数指令一览表

函　数	功　能
ABS(X)	对 X 取绝对值
SGN(X)	判别 X 的符号
RND(X)	产生 X 以内的随机整数
ASC(字符)	将字符转换成 ASCII 码
CHR $ (X)	求 ASCII 码 X 对应的字符
VAL(字符)	将字符串转换成 10 进制数
STR $ (X)	将 10 进制数转换成字符串
HEX $ (X)	将 10 进制数转换成 16 进制数
LEFT $ (字符串,n)	从字符串左边取 n 个字符
RIGHT $ (字符串,n)	从字符串右边取 n 个字符
MID $ (字符串,N,n)	从字符串中的第 N 位取 n 个字符
INSTR(字符串,子串)	检索,子串在字符串中的位置
LEN(字符串)	计算字符串的长度

14—4 特殊函数指令一览表

函　数	功　能
PEEK(X)	取指定内存地址 X 中的数值
POS(0)	了解光标在目前画面上的水平位置
CSRLIN	给出光标在垂直方向的位置座标
FRE	测试内存单元的空余数
STICK(n)	读操纵器的方向键值
STRIG(n)	读操纵器的功能键值

14—4 （续）

函　　数	功　　能
SCR$(X,Y,SW)	求出BG面(X,Y)处图案的数值
INKEY$(0)	读取键值输入的一个值
TAB	（待定）
SPC	（待定）

14—5 画面控制指令一览表

指　　令	功　　能
LOCATE X,Y	设定显示位置的座标(X,Y)
COLOR X,Y,N	在屏幕上(X,Y)位置染n号颜色
CGEN (n)	设定BG、SP面上符号的分配
CGSET[m][,n]	规定BG、SP面上的颜色组合
CLS	清屏
PALETB n,c1,c2,c3,c4	定义BG面的颜色代码
PALETS n,c1,c2,c3,c4	定义SP面的颜色代码
VIEW	调出BG GRAPHIC画面
DEF MOVE (n) = SPRITE(A,B,C,D,E,F)	定义动作SP
MOVE n0[,n1,……]	启动SP运动
CUT n0[,n1,……]	使运动中的SP停止
ERA n0[,n1,……]	令运动中的SP停止并消失
POSITION n,X,Y	规定n号动作SP的起点座标(X,Y)
XPOS(n)	求出n号动作SP的水平座标
YPOS(n)	求出n号动作SP的垂直座标
VCT(n)	求出n号动作SP的运动方向
CRASH(n)	判断n号动作SP是否与其它SP相遇
MOVE(n)	判断n号动作SP的动作是否结束

14—5 （续）

指　令	功　能
CAN n0[,n1,……]	解除动作 SP 的定义
DEF SPRITE n,(A,B,C,D,E)=字符表达式	定义 n 号显示 SP
SPRITE n,X,Y	定义 n 号显示 SP 的显示座标(X,Y)
SPRITE n	令 n 号显示 SP 消失
SPRITE ON	开 SP 显示
SPRITE OFF	关 SP 显示

14—6　出错处理指令一览表

指　令	功　能
ON ERROR GOTO n	设置出错转移入口
ERL	给出最近一次出错语句所在的行号
ERR	给出最近一次出错类型的代码
ERROR n	定义用户自己的出借代码
RESUME [NEXT/行号]	从出错处理子程序返回原程序

附录 15　音名与频率对照表

八度音	音名	频　率(Hz)	八度音	音名	频　率(Hz)
O0	C	55		F	293.66
	C#	58.27		F#	311.13
	D	61.735		G	329.63
	D#	65.406		G#	349.23
	E	69.296		A	369.99
	F	73.416		A#	392
	F#	77.782		B	415.3
	G	82.407	O3	C	440
	F#	87.307		C#	466.16
	A	92.499		D	493.88
	A#	97.999		D#	523.25
	B	103.83		E	554.37
O1	C	110		F	587.33
	C#	116.54		F#	622.25
	D	123.47		G	659.25
	D#	130.81		G#	698.46
	E	138.59		A	740
	F	146.83		A#	784
	F#	155.56		B	830.61
	G	164.81	O4	C	880
	G#	174.61		C#	932.33
	A	185		D	987.77
	A#	196		D#	1046.5
	B	207.65		E	1108.7
O2	C	220		F	1174.7
	C#	233.08	O5	C#	1864.7
	D	246.94		D	1975.5
	D#	261.63		D#	2093
	E	277.18		E	2217.5

附录 15 （续）

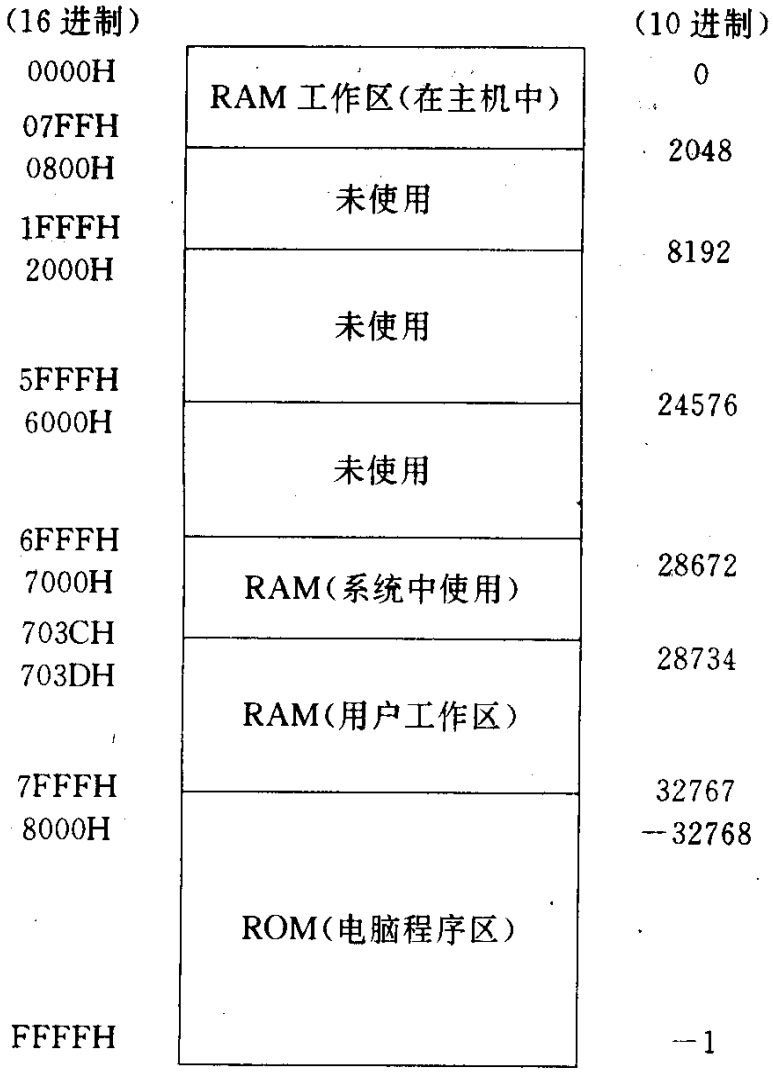
八度音	音名	频　率(Hz)	八度音	音名	频　率(Hz)
	$F^{\#}$	1244.5	O5	F	2349.3
	G	1318.5		$F^{\#}$	2489
	$G^{\#}$	1396.9		G	2637
	A	1480		$G^{\#}$	2793.8
	$A^{\#}$	1568		A	2960
	B	1661.2		$A^{\#}$	3136
O5	C	1760		B	3322.4

注：本表各频率值是根据 12 平均律关系计算出来的，与实际稍有出入，仅供参考。

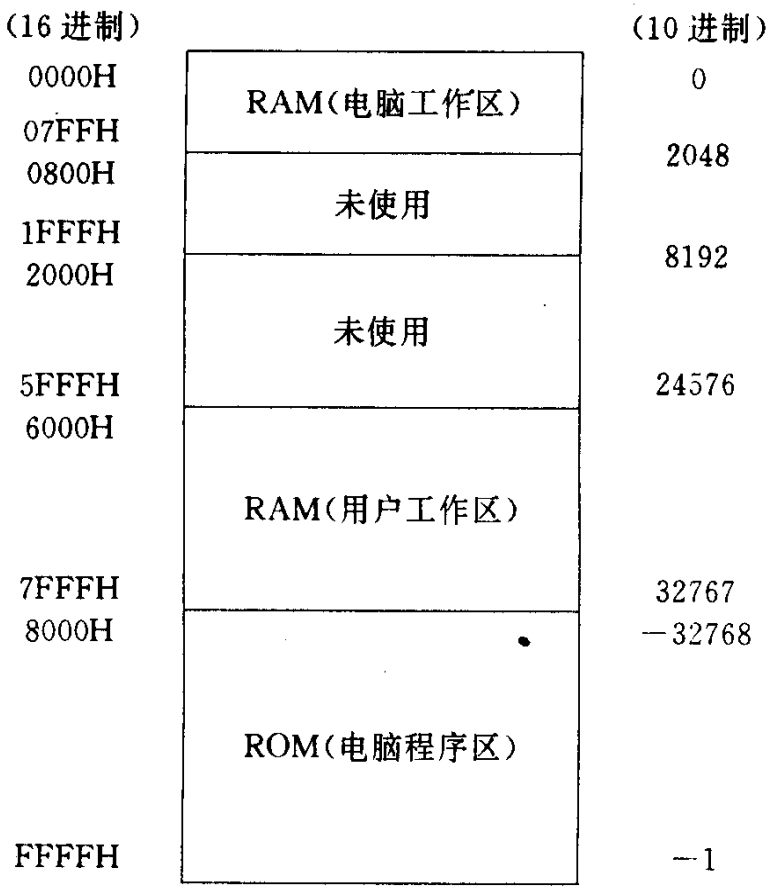
附录 16　音符时值表

数字	音　符	节　拍	示　　例
9	全音符	4 拍	1－－－
8	副 2 分音符	3 拍	1－－
7	2 分音符	2 拍	1－
6	副 4 分符	$1\frac{1}{2}$拍	1·
5	4 分音符	1 拍	1
4	副 8 分音符	3/4 拍	$\underline{1}$·
3	8 分音符	1/2 拍	$\underline{1}$
2	副 16 分音符	3/8 拍	$\underline{\underline{1}}$·
1	16 分音符	1/4 拍	$\underline{\underline{1}}$
0	32 分音符	1/8 拍	$\underline{\underline{\underline{1}}}$

附录 17　BS. 2A 内存分配表



附录 18　V3.10 内存分配表



附录 19　按键代码表

按键名称	10 进制代码	16 进制代码	按键名称	10 进制代码	16 进制代码
STOP	1	01			
ESC	27	1B	Ctrl+I	9	09
Tab	9	09	Ctrl+L	12	0C
Home	11	0B	Ctrl+M	13	0D
Del	8	08	Ctrl+N	14	0E
Ins	18	12	Ctrl+O	15	0F
Enter	13	0D	Ctrl+P	16	10
↑	30	1E	Ctrl+Q	17	11
↓	31	1F	Ctrl+R	18	12
←	29	1D	Ctrl+S	19	13
→	28	1C	Ctrl+T	20	14
Home+Shift	12	0C	Ctrl+U	21	15
Ctrl+C	3	03	Ctrl+V	22	16
Ctrl+B	2	02	Ctrl+W	23	17
Ctrl+E	5	05	Ctrl+X	24	18
Ctrl+F	6	06	Ctrl+Y	25	19
Ctrl+G	7	07	Ctrl+Z	26	1A

注：Ctrl+L=Home+Shift，Ctrl+M=Enter，Ctrl+R=Ins

后　记

我的文章自一九九一年在《电子与电脑》杂志连载以来，收到数百封读者来信，提出了各种各样的问题。这些问题的70%可以在本书中找到圆满的答案，但尚有一些问题诸如：任天堂游戏的画面与背景面一样大，可以达到30行×32列，而为什么F BASIC的背景画面仅有24行×28列，任天堂游戏中有各种各样的卡通，有千奇百怪的音响效果和美妙动听的背景音乐，背景画面可以左右卷绕或上下滚动，而且背景的卷绕、滚动与音响、卡通的运动控制可以同步进行，F BASIC为什么不能，F BASIC能否实现等问题得不到解答。因而，阅毕本书后会感到不满足，会提出本书是否还有续集出版的问题。

在此，我可以肯定地告诉大家：F BASIC是一种面向游戏的高级语言，因而任天堂游戏中的各种效果都可以在F BASIC中模拟或实现（当然有些效果的实现要靠编写机器语言子程序）；本书还有姊妹篇："F BASIC编程特技的深入研究"，书中将详细分析F BASIC的工作系统软件，通过大量调用系统子程序，以较多的示例、较小的程序量，模拟实现任天堂游戏中的各种效果。当阅读了该书后，每一位朋友都将可以大言不惭地说：我已经真正掌握了F BASIC。

编　者

1992.8